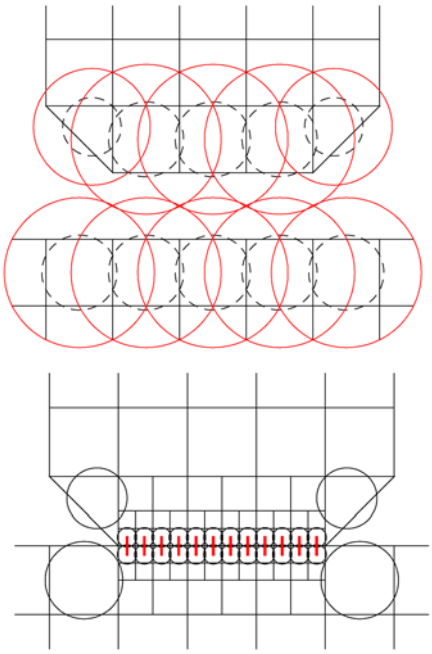
European Commission

JRC TECHNICAL REPORTS

Generalization of the pinball contact/impact model for use with mesh adaptivity and element erosion in EUROPLEXUS

Folco Casadei
Vegard Aune
Georgios Valsamos
Martin Larcher
2016



epx europlexus software

EUR 27888 EN

Joint Research Centre

Generalization of the pinball contact/impact model for use with mesh adaptivity and element erosion in EUROPLEXUS

JRC Science Hub
https://ec.europa.eu/jrc

JRC101013

EUR 27888 EN

ISBN 978-92-79-57972-1

ISSN 1831-9424

doi:10.2788/333017

Printed in Italy

How to cite: Authors; title; EUR; doi

Generalization of the pinball contact/impact model for use with mesh adaptivity and element erosion in EUROPLEXUS

F. Casadei[1], V. Aune[2], G. Valsamos[3], and M. Larcher[3]

[1] *Retired from JRC ELSA*
[2] *Structural Impact Laboratory (SIMLab), Centre for Research-based Innovation (CRI) and Department of Structural Engineering, Norwegian University of Science and Technology, Rich. Birkelands vei 1A, NO-7491 Trondheim, Norway*
[3] *European Laboratory for Structural Assessment (ELSA), Institute for the Protection and Security of the Citizen (IPSC) Joint Research Centre (JRC), 21027 Ispra, Italy*

March 20, 2016

Contents

List of Tables

List of Figures

List of Algorithms

1 Introduction

This report presents the generalization of the pinball-based contact-impact model (`PINB`) of the EUROPLEXUS code for use in conjunction with mesh adaptivity, i.e. with adaptive mesh refinement and un-refinement. The interaction of the pinball model with the mechanism of element failure and erosion is also revised.

EUROPLEXUS [1] (also abbreviated as EPX) is a computer code jointly developed by the French Commissariat à l'Energie Atomique (CEA DMT Saclay) and by EC-JRC. The code application domain is the numerical simulation of fast transient phenomena such as explosions, crashes and impacts in complex three-dimensional fluid-structure systems. The Cast3m [2] software from CEA is used as a pre-processor to EPX when it is necessary to generate complex meshes.

1.1 Contact

The most popular contact algorithms available in Finite Element (FE) computer codes are probably the so-called slide line (in 2D) and slide surface (in 3D) algorithms proposed by Hallquist and Benson [3, 4]. They are based on the notion of penetration of slave nodes into master segments (in 2D) or into master surfaces (in 3D). These algorithms suffer from a number of geometrically pathological cases in which physical penetration is not detected.

The pinball method proposed by Belytschko and co-workers from the late 80's [5–12] for application in impact problems with perforation is much more robust as concerns penetration detection. The pinball contact-impact method has been implemented in EPX in [14–18], initially based on strong coupling via a Lagrange-multiplier (LM) based solution strategy of the contact constraints (see [13] for details of the method) and more recently by using weak coupling based on a penalty approach, see [19]. The latter report also contains a description of the implementation of Assembled Surface Normals (ASNs) in the pinball model, by an algorithm inspired to the one proposed by Belytschko in reference [5].

The original pinball model of Belytschko and its implementation in EPX (`PINB` keyword) are based upon spherical pinballs. This approach is extremely robust and efficient in detecting penetration, but suffers from certain drawbacks when it comes to imposing contact constraints.

When dealing with slender elements (highly deformed continuum elements or structural members such as bars, beams and shells) the basic pinball method which associates a sphere with each element is no longer applicable and a hierarchic pinball method, consisting of splitting each penetrating pinball into a series of smaller pinballs (recursively, until a certain minimum size is reached) must be used instead, to enhance the resolution of penetration detection. Apart from the complexity and relative inefficiency of a recursive approach, difficulties arise when imposing multiple contacts between sub-pinballs with the LM method, because in this case redundant constraints may be generated and the system of constraints may become singular (see e.g. Section 7 of reference [15]). Various techniques have been devised in order to get rid of redundant constraints, but this is a complex and inefficient task, very difficult to perform in general terms. The penalty approach does not suffer from this limitation, but it needs some tuning parameters, which render solutions more laborious (besides being non-unique) and less reliable for the user, than the LM method.

Another difficulty in dealing with spherical-only pinballs is the determination of the local contact direction. The simplistic approach of using the line joining the two contacting pinball centres is sufficient in some cases (e.g. in fast impact with perforation) but may introduce large errors in other

situations (such as sliding-dominated contact between smooth bodies). Recent (re-)developmeent of the ASN technique (see [19]) which associates a unique normal direction with each pinball or subpinball and then introduces rules for computing a "better" contact direction than the simple centresjoining line can alleviate these problems in many, but not in all, cases.

In a still ongoing work [20] we explore the possibility of using "generalized" pinballs (GPINs), of various geometrical shapes (not only spherical) in an attempt to avoid the problems highlighted above (especially those affecting the LM method), while retaining as far as possible the robustness and simplicity of the original pinball approach. The use of a variety of shapes allows to get rid of the necessity of a hierarchic procedure at the expense, however, of (slightly) more complex penetration checks than with spheres. This should ensure that no (or fewer) redundant constraints appear and therefore their elimination is no longer necessary, with positive effects also on the treatment of rebound.

1.2 Adaptivity

In recent years (from 2010 on) mesh adaptivity (or AMR for Adaptive Mesh Refinement) has been implemented in EPX. Automatic mesh adaptation is now available in continuum as well as in structural element types, and in solid as well as in fluid materials, including both Finite Element (FE) and Finite Volume (FV) formulations. Details of adaptivity developments in EPX can be found in references [21–39].

Reference [21] presented the first implementation in EPX of an adaptive mesh refinement and un-refinement procedure, in two space dimensions (element shape QUA4) for solid mechanics. The procedure was extended to fluid mechanics (FE formulation) in 2D in reference [22]. Then, reference [23] applied a similar refinement and un-refinement procedure in three space dimensions to the CUB8 element shape, both in solid mechanics and in fluid mechanics (FE formulation).

All numerical examples presented in references [21–23] with a variable mesh used a so-called "manual" mesh adaptation directive, the `WAVE` directive (see the code manual in reference [1]), first introduced in reference [21]. This directive refines the mesh along "wavefronts" that are specified by the user, e.g. according to a known analytical solution to the problem considered. This technique was used with success to simulate a bar problem (in solid mechanics) and a shock tube problem (in fluid mechanics) both in 2D and in 3D [21–23].

However, those solutions cannot be qualified as "true" adaptive solutions, because in (true) adaptivity mesh refinement and un-refinement should be completely automatic, based upon suitable error estimators or error indicators. The formulation of error estimators in fast transient dynamics is challenging and is still a subject of research. The use of so-called error indicators, however, is much simpler.

For this reason, subsequent work in EPX focused on error indicators. References [24] and [25] document a first prototype implementation of adaptivity based upon error indicators in EPX, limited to 2D problems in continuum and fluid mechanics. An extension of the indicator technique to 3D is under development but has not been completed and documented yet.

Publications [26, 27] focus on the natural quantities of interest in goal-oriented error assessment and adaptivity, but limited to the case of linear elasto-dynamics.

The adaptive technique was then applied to Cell-Centred Finite Volumes (CCFV) for the description of the fluid domain, first in 2D (see [28]) and then also in 3D [29]. More recently, the technique has also been extended for use with the CDEM combustion model which makes use of the CCFV formulation [30].

A complete description of the element refinement and un-refinement techniques used in mesh adaptation has been published in a journal paper [31].

A first contribution towards combination of mesh adaptivity with Fluid-Structure Interaction (FSI) was given in reference [32], in which a model is described that automatically refines the fluid mesh in the vicinity of an embedded structure which can move and deform until and beyond rupture (but without being itself subjected to adaptivity).

In [33] adaptivity was activated for simplex elements (triangles in 2D and tetrahedra in 3D). The report [34] extends adaptivity to CEA's family of fluid elements. Finally, reference [35] extends adaptivity to shell, beam and bar structural elements. It becomes therefore possible to have mesh

adaptivity both in a fluid and at the same time in a structure (typically made of shells) embedded in the fluid.

Combined criteria for adaptive mesh refinement were addressed in [36], while [37] introduced simultaneous adaptation of both fluid and structural mesh in FSI calculations.

Report [38] dealt with the interpretation of mesh-adaptive results and some peculiarities that arise in the post-processing of such calculations. Finally, [39] proposed a decoupled formulation of constraints on hanging nodes in adaptivity which sometimes allows to greatly reduce CPU time in this types of simulations.

1.3 Element failure and erosion

Element failure and element erosion are treated as two separate models in EPX, although they are of course inter-related.

Failure usually occurs at element Gauss (or integration) points, when the material model has reached a certain threshold, e.g. a level of damage beyond which it provides no further resistance. The material failure criterion often depends on the particular material model. Each Gauss point has its own stress / strain history and its own set of hardening parameters, so it may fail individually. Upon failure, the resistance of material at the concerned Gauss point is usually completely removed and therefore the load will increase on the neighboring points due to re-distribution of equilibrium, possibly leading to a chain failure. A failed Gauss point is usually not allowed to recover any resistance, so it remains failed for the rest of the computation.

Once a certain fraction of an element's Gauss points have failed, the complete element is flagged as failed and removed from the computation. The percentage of failed Gauss points for this to occur may be set from the user and varies from 0 (brittlest behaviour: the element fails as soon as any of its Gauss points fails) to 1 (toughest behaviour: the element fails when all of its Gauss points have failed).

Other (more phenomenological) failure models are available, which act on the element as a whole. For example, a criterion which states failure of a (brittle) glass panel when a certain displacement is reached.

The two mechanisms of failure and erosion are kept separated in the input declaration in order to achieve fine-grain control over the propagation of structure degradation. Of course in most (but not in all) cases, element erosion is a direct consequence of (and is piloted by) element failure.

Element failure / erosion is obviously interconnected with adaptive mesh refinement. For example, if an element is refined and then some of its descendents fail and are eroded, then the element will no longer be suitable for un-refinement.

The present report deals mainly with the combination of mesh adaptivity with contact treated by the `PINB` method in the same calculation. It also reviews the most common aspects of interaction between mesh adaptivity and element failure / erosion. The document is organized as follows. Section 2 presents the proposed formulation. Section 3 shows some numerical examples, Section 4 gives some conclusions and suggestions for future work. All the input files mentioned in the present report are listed in the Appendix.

2 Combination of adaptivity with pinball contact

Combining contact with mesh adaptivity is a potentially vast subject. At least two types of interaction (or relationship) between AMR and contact can be envisaged.

In the *first case*, adaptivity is triggered by contact. We refer to this as *contact-driven adaptivity*. For example, imagine two approaching bodies which are about to enter in contact with each other. In order to increase the accuracy and resolution of contact treatment, it is desirable that the mesh be particularly fine near the contacting surfaces. Therefore, adaptivity can be triggered by the contact model itself, in order to locally refine the mesh just before contact first occurs.

In the *second case*, adaptivity is triggered by a different criterion than contact. We refer to this as *adaptivity-driven contact*. In this case adaptivity is piloted by an error indicator or by a damage-based

threshold criterion, to name just a few of the possibilities available in EPX. If contact is foreseen (by means of embedded pinballs) in some of the elements that are being refined, then the descendents of such elements must somehow inherit the contact capabilities of their ancestors. In other words, descendents of pinball-endowed elements must also be suitably endowed by (smaller) pinballs.

2.1 Contact-driven adaptivity by point clouds

The first type of relationship has been already addressed in EPX by a pilot development (undocumented) performed at CEA and based upon the notion of *point clouds* (PCLD). The development is limited to (and currently works only with) the parallel, MPI-based version of EPX. An example of this model (from the standard suite of non-regression tests of EPX) is test BM_MPI_ADAP_PINBALLS, whose input file is listed in the Appendix.

This example is illustrated in Figure 1. In 1(a) the upper body is moving downwards at uniform initial velocity towards the lower body, which is blocked at its base. Both bodies are endowed with (0-level or parent) pinballs on their elements located along the respective outer surfaces. These are represented by small dots in the drawings. Both bodies use the same elastic material. In 1(b) the distance between the two bodies has diminished so that the contact algorithm starts to refine the mesh. New (parent) pinballs are automatically inserted in the created descendent elements which are potentially subjected to contact. The refinement process continues in 1(c), when contact first occurs. In 1(d) mesh refinement is "spread" over the bodies by the error indicator (based upon the curvature of the velocity field), which follows the propagating stress wave fronts generated by the contact. In 1(e) stress waves have spread over the entire bodies and some reflections have also occurred, so that rebound starts taking place. The picture in 1(f) shows the final configuration. Some elastic waves are still present in the rebounding bodies, so the mesh is not completely unrefined yet.

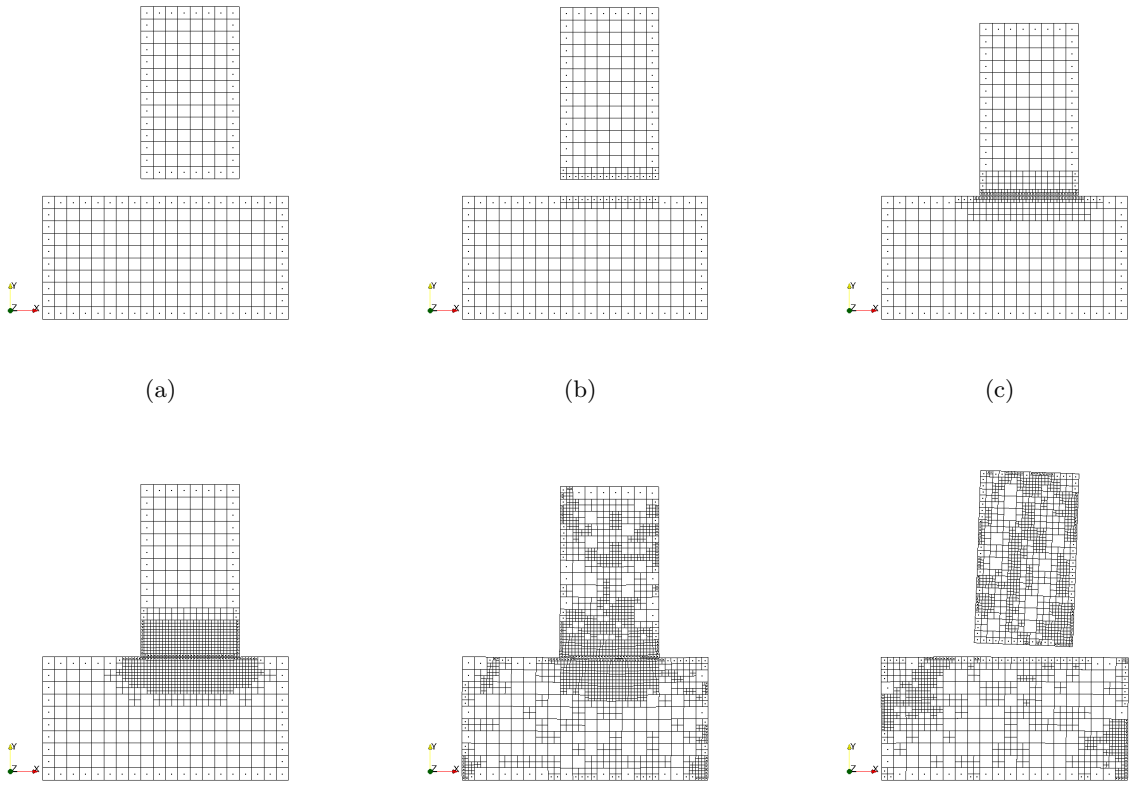


Figure 1: Contact-driven adaptivity in test case BM_MPI_ADAP_PINBALLS.

2.2 Adaptivity-driven contact

The second type of relationship between adaptivity and contact is the subject of the present development. In this case, mesh refinement and un-refinement is decided by a mechanism *independent* from contact. The scope is then to *propagate* the contact model (i.e. the pinballs) to the newly created descendent elements.

2.2.1 Rules for pinballs with adaptivity

The following rule related to element refinement in adaptivity-driven contact may be postulated.
Rule 1. When an element containing a (parent) pinball is *refined* (split), its descendents must *inherit* (parent) pinballs with characteristics suitably derived from those of their parent element's pinball.

This rule is of course recursive, i.e. it applies (recursively) to any level of mesh refinement. A rule must also be provided to govern mesh unrefinement.
Rule 2. When an element containing (parent) pinballs is *unrefined* its descendent elements are destroyed, so any (parent) pinballs associated with such descendents must be destroyed as well.

2.2.2 Implementation details of standard model

The implementation of pinballs in EPX is based upon two Fortran90 derived types located in module M_PINBALL. The first derived type (PINBALL) describes a parent (0-level) pinball:

```
TYPE PINBALL ! "PARENT" PINBALL DEFINITION
  INTEGER :: ELEMENT     ! ELEMENT INDEX
  REAL(8) :: RADIUS      ! RADIUS OF THE PINBALL
  REAL(8) :: CENTER(3)   ! CENTER OF THE PINBALL
  INTEGER :: MAXLEV      ! MAXIMUM LEVEL OF DESCENDENT PINBALLS
  INTEGER :: SET         ! SET TO WHICH PINBALL BELONGS
                         ! (<0 IF SELF-CONTACT ENABLED)
  INTEGER :: HARDNESS    ! ASSOCIATED "HARDNESS" (ONLY FOR FLAT
                         ! CONTACT)
  REAL(8) :: DTPINB      ! LIMIT TIME STEP ON THIS PINBALL'S
                         ! ELEMENT AS DICTATED BY PCONTACTS
                         ! (0 IF NONE)
  LOGICAL :: IS_ACTIVE   ! ACTIVE PINBALL OR NOT
  REAL(8) :: ASN(4)      ! ASN OF THE PINBALL:
                         ! (1)  = LENGTH OF THE ASN BEFORE
                         !        NORMALIZATION (>= 0.0)
                         ! (2:) = COMPONENTS OF THE NORMALIZED
                         !        ASN (IDIM VALUES) OF LENGTH 1.0
                         !        (IF ASN(1)>0) OR 0.0 (IF ASN(1)=0)
END TYPE PINBALL
```

The second derived type (DESCENDENT_PINBALL) describes a descendent (higher-level) pinball:

```
TYPE DESCENDENT_PINBALL ! "DESCENDENT" PINBALL DEFINITION
  INTEGER :: ANCESTOR    ! INDEX OF ANCESTOR (0-LEVEL) PINBALL
  INTEGER :: LEVEL       ! LEVEL OF THIS PINBALL (>= 0)
  REAL(8), POINTER :: XYZ(:,:) ! COORDS OF "NODES" OF PINBALL
  REAL(8) :: RADIUS      ! RADIUS OF THE PINBALL
  REAL(8) :: CENTER(3)   ! CENTER OF THE PINBALL
  INTEGER, POINTER :: IFACE(:) ! FACE INDEXES
                           !   FOR CONTINUUM ELEMENTS:
                           ! N>0=EXTERNAL, ON FACE N OF PARENT
                           ! M<0=INTERNAL, SEES PAR. NEIGHBOUR M
                           !   0=INTERNAL (FROM CUT OF PARENT)
                           !   FOR BEAM/SHELL ELEMENTS:
                           ! SPECIAL CONVENTION (SEE REPORT)
  REAL(8) :: ASN(4)      ! ASN OF THE DESCENDENT PINBALL:
                         ! (1)  = LENGTH OF THE ASN BEFORE
                         !        NORMALIZATION (ALWAYS > 0.0)
                         ! (2:) = COMPONENTS OF THE NORMALIZED
                         !        ASN (IDIM VALUES) OF LENGTH 1.0
END TYPE DESCENDENT_PINBALL
```

Module M_PINBALLS_DATA contains many additional data, among which the most notable are:

```
INTEGER :: N_PINBALLS ! TOTAL NUMBER OF PARENT (0-LEVEL) PINBALLS
TYPE(PINBALL), POINTER :: PINBALLS(:) ! ARRAY OF PARENT PINBALLS
```

The total number of parent pinballs (`N_PINBALLS`) is static (constant) in EPX and is automatically determined upon reading of the `PINB` directive in the EPX input file. The list of parent pinballs (array `PINBALLS`) is built automatically immediately after, before starting the transient calculation. No dimensioning concerning pinballs is necessary. The list of parent pinballs is never updated during a calculation.

If a *hierarchic* pinball method is selected by the user (either directly by keywords `MLEV` or `DMIN` in the input file, or indirectly by associating pinballs with shell elements), then descendent pinballs are generated in order to increase the resolution and accuracy of contact detection. Descendent pinballs are automatically and recursively created as long as contact (penetration) between (lower-level) pinballs is detected, and this up to the highest prescribed level. Only the contacts associated with the descendent pinballs belonging to the highest specified level are retained. The information about one such contact is stored in a derived type `PCONTACT`, which is defined in module M_PCONTACT:

```
TYPE PCONTACT
  INTEGER :: PA, PB      ! INDEXES OF THE TWO (PARENT) PINBALLS
  REAL(8) :: NAB(3)      ! UNIT NORMAL FROM A TO B
                         ! "JOINS CENTERS"    IF PINBALL_FNOR = 0,
                         ! "MEAN" NORMAL      IF PINBALL_FNOR = 1,
                         ! "COMMON" NORMAL    IF PINBALL_FNOR = 2,
                         ! "ASN" BUILT NORMAL IF PINBALL_FNOR = 3
  REAL(8), POINTER :: SHA(:), SHB(:)
                         ! SHAPE FUNCTIONS OF ELEMENT A AND B AT
                         ! INTERACTION POINTS A' AND B'
  REAL(8) :: CSIA(3), CSIB(3) ! NORMALIZED COORS OF CONTACT POINTS
                         ! WITH RESPECT TO PARENT ELEMENTS
  TYPE(DESCENDENT_PINBALL) :: DPI, DPJ ! DESCENDENT PINBALLS IN
                         ! CONTACT (WITH LEVEL=0 IF PARENT)
                         ! THESE ARE COPIES (NOT POINTERS)
  LOGICAL :: HAS_COMMON_NORMAL ! IS ASSOCIATED NORMAL "COMMON"?
  INTEGER :: NODE1, NODE2 ! NODES OF THE CONTACT (EACH MAY BE 0)
                          ! N1 N2  --> NN CONSTRAINT
                          !  0  0  --> PP CONSTRAINT
                          ! N1  0  OR
                          !  0 N2  --> NP CONSTRAINT
  INTEGER :: ELEM1, ELEM2 ! ELEMENTS OF THE CONTACT (EACH MAY BE 0)
                          !  0  0  --> NN CONSTRAINT
                          ! E1 E2  --> PP CONSTRAINT
                          !  0 E2  OR
                          ! E1  0  --> NP CONSTRAINT
  LOGICAL :: FLAG         ! GENERIC FLAG FOR VARIOUS OPERATIONS
  REAL(8) :: PENETR       ! PENETRATION (ALONG NAB) IF USEFUL
  REAL(8) :: PDOT         ! PENETRATION RATE (ALONG NAB) IF USEFUL
  INTEGER :: TYPEI, TYPEJ ! TYPE OF CONTACTING (SUB-)PINBALL:
                          ! 0 = ELEMENT (PARENT) PINBALL
                          ! 1 = VERTEX SUB-PINBALL
                          ! 2 = CORNER SUB-PINBALL (3D ONLY)
                          ! 3 = FACE SUB-PINBALL
  INTEGER :: NODEI(5), NODEJ(5) ! N. OF CONTACTING SUB-PINBALL NODES,
                          ! THEN LIST OF THESE NODES (GLOBAL INDEXES):
                          ! 1 NODE FOR A VERTEX SUB-PINBALL
                          ! 2 NODES FOR A CORNER SUB-PINBALL (3D ONLY)
                          ! 2, 3 OR 4 NODES FOR A FACE SUB-PINBALL
END TYPE PCONTACT
```

The contacts found in the current time step of the calculation are stored in `PCONTACTS`, a doubly-linked list of derived types `PCONTACT`, which is defined in module M_PINBALLS_DATA:

```
TYPE(BILINKED_LIST_PCONTACT) :: PCONTACTS
```

The linked data structure is necessary (as opposed to an array, for example) because the number of potential contacts is unknown and hardly predictable.

For the scope of the current development, it is important to note that the descendent pinballs (and the list of pinball contacts) are totally dynamic. They are automatically created at the beginning of each time step and completely destroyed at the end of the step after the contact forces have been

computed. Also, no list of *descendent* pinballs is ever built. Only the *parent* pinballs and their list (array PINBALLS) are kept throughout the calculation (and saved/restored in case of calculation restart).

2.2.3 Strategy for the adaptive model

To cope with adaptivity, the following modifications are foreseen in the data structure.

First, some dimensioning related to pinballs is added in the directive related to adaptivity dimensions DIME ADAP. The new syntax is:

```
DIME ... ADAP ... NPIN npin ... ENDA ... TERM
```

where npin is the (maximum) number of *parent* pinballs that will be possibly created during the adaptive mesh refinement process. This quantity is stored in a new variable N_PINBALLS_EXT, i.e. the (maximum) number of (parent) pinballs in the "extension" zone, which is added to the module M_PINBALLS_DATA. In the same module, the former variable N_PINBALLS is renamed N_PINBALLS_BASE, i.e. the number of (parent) pinballs contained in the *base* mesh. Then, the total number of (parent) pinballs N_PINBALLS is re-defined as the sum of N_PINBALLS_BASE and N_PINBALLS_EXT. A list of parent pinballs is created containing first all the N_PINBALLS_BASE parent pinballs and then the N_PINBALLS_EXT pinballs in the extension zone. Initially, the extension pinballs are all "empty": their associated element ELEMENT is set to 0 and their activity flag IS_ACTIVE is set to .FALSE.. To summarize:

```
INTEGER :: N_PINBALLS_BASE ! TOTAL NUMBER OF BASE PARENT (0-LEVEL) PINBALLS
INTEGER :: N_PINBALLS_EXT  ! TOTAL NUMBER OF EXTENSION PARENT (0-LEVEL) PINBALLS
INTEGER :: N_PINBALLS ! TOTAL NUMBER OF PARENT (0-LEVEL) PINBALLS
                      ! (= N_PINBALLS_BASE + N_PINBALLS_EXT)
TYPE(PINBALL), POINTER :: PINBALLS(:) ! ARRAY OF PARENT PINBALLS
```

The pinballs in the extension zone get used (in ascending order) as the mesh refinement proceeds. In module M_ADAPTIVITY_TREE a new subroutine GET_FREE_PINB_ADAP is added (modelled after the similar routine GET_FREE_ELEM_ADAP used for elements) which returns the index of the first "free" (i.e. not used) pinball in the extension zone (or 0 if the extension zone is exhausted). When invoked, the routine updates the index of the first unused extension pinball FIRST_FREE_PINB_ADAP (added in M_ADAPTIVITY_DATA) by searching forward along the PINBALLS array. Further variables FIRST_PINB_ADAP (simply equal to N_PINBALLS_BASE+1) and MAX_USED_PINB_ADAP (the maximum pinball index ever used in the calculation so far, which can be used after a calculation has been completed in order to set the pinballs-related dimensioning npin to the minimum admissible value) are also added to the module M_ADAPTIVITY_DATA:

```
INTEGER :: FIRST_FREE_PINB_ADAP ! FIRST FREE PINBALL IN THE EXTENSION ZONE
INTEGER :: FIRST_PINB_ADAP ! FIRST PINBALL IN THE EXTENSION ZONE (=N_PINBALLS_BASE+1)
INTEGER :: MAX_USED_PINB_ADAP ! MAX INDEX OF PINBALL USED SO FAR
```

2.2.4 Pinballs activation algorithm

The activation status of each (parent) pinball (IS_ACIVE flag in the derived type PINBALL) determines whether or not the pinball is a candidate for contact (interpenetration) checks with other pinballs at each given moment of the computation.

This status may change during the course of the computation. For example, an active pinball may become inactive when the associated element fails and is eroded. Conversely, an inactive pinball, embedded within the body of a continuum and (initially) not on its surface may become active when, due to element erosion, the associated element finds itself on the (current) surface of the body. The rules for pinball activation are stated next.

During the **initialization** phase, i.e. right after reading the PINB directive(s):

- All (parent) pinballs associated with flying debris (particle) elements (DEBR) which are initially inactive, are flagged as inactive. Recall that in EPX's flying debris model, inactive debris

particles are embedded in the elements which are candidate for failure. As one such element fails and is eroded, the debris particles are activated and provide an approximated representation of the structural (micro) fragments. Pinballs may be attached to such fragments so that their impact on the surrounding structures is modelled.

- All other (parent) "base" pinballs are flagged as active. Base pinballs are those associated with base elements in an adaptive computation (or simply with all elements in a non-adaptive computation)
- Any (parent) "extension" pinballs in a mesh adaptive calculation are flagged as inactive.

Then, at **each time step** of the calculation (including step 0, i.e. at the initial time), a routine UPDATE_PINBALLS_ACTIVATION (formerly named ACTIVATE_PINBALLS) is invoked in order to update the status of each (parent) pinball, and this *before* computing pinball interpenetrations and contact forces. The algorithm for parent pinball status updating is as follows:

1. Loop over all (parent) pinballs. Let p be the current pinball and e the associated element.
2. If $e = 0$, then p (necessarily in the extension zone) is currently unused, so it must be inactive. We check that p is indeed inactive, for consistency, then move on to the next pinball.
3. Else $e > 0$. If the case is adaptive, we check the adaptivity status of element e. If e has children (i.e. it is currently inactive in adaptivity), then p must be inactive as well (pinball deactivation should occur as part of the element adaptive splitting process). We check that p is indeed inactive, for consistency, then move on to the next pinball.
4. Else $e > 0$ and e has no children. We check the erosion status of element e. If e is currently eroded, then p must be inactive as well (pinball deactivation should occur as part of the element erosion process). We check that p is indeed inactive, for consistency, then move on to the next pinball.
5. Else e is currently active (in case of adaptivity) and is currently not eroded. We inspect the type T of element e.
   - If ($T = 140$) then e is a debris (`DEBR`) element. If the debris element is currently active, the associated pinball p must also be active, else p must be inactive. Pinball activation must have been done as part of the debris activation process. We check that the status of p is indeed the expected one, for consistency, then move on to the next pinball.
   - Else if ($T = 79$) then e is a fluid SPH particle (`BILL`) element. If there is an explosive `JWLS` material in the model and e has not yet been reached by the detonation front (this is indicated by the element-based array FAILURE_LEVEL), then p is flagged as inactive. Else, p is flagged as active. We move on to the next pinball.
   - Else if T is such that e is a "continuum" element (not a "strucural" one, such as a shell, plate, beam or bar element), then we inspect the element's *faces*: if at least one face is *free* (see definition below), e is on the current external surface of the body and p is flagged as active, else e is currently internal to the body and p is flagged as inactive. We move on to the next pinball.
   - Else T is such that e is not a "continuum" element, despite being a solid element (in fact, it would not make sense to associate pinballs with fluid elements, except in the special case of SPH fluid particles already considered above). For example, e may be a structural element (shell, plate, beam, bar) or a material point (e.g. a `PMAT`). For these elements, an associated pinball is always active (since, unlike for continuum, such elements are always "on the surface" of the body) as long as the element itself is active, i.e. the element is not eroded (which must be the case here, due to the above checks) and, in case of adaptivity, the element has no children. For these elements we let the status of p unchanged, after

checking it for consistency: if the model is adaptive and e has children, then e is inactive and p must be inactive as well (this must have been set as part of the element splitting procedure), else e is active and p must be active as well.

6. If (parent) pinballs are not exhausted, loop on to the next pinball, else exit.

Note that this algorithm takes into account two different effects of element erosion on pinball contact. The first (direct) effect is at point 3 of the algorithm for an eroded element itself. The second (indirect) effect is at point 5 of the algorithm, when it deals with continuum elements (third sub-case). This accounts for the effect that an element's erosion has *on its neighboring elements*, which become located on the (current) external surface of the body and therefore become candidates for contact.

The above algorithm uses the notion of **free face** for a continuum element. In 2D, faces are 2-node segments while in 3D faces are either 3-node triangles or 4-node quadrilaterals. The definition is as follows (we distinguish between a non-adaptive and an adaptive case for simplicity).
In a **non-adaptive** case:

1. The face f of a continuum element e is *not free* if e has a neighbor element ν of continuum (not `CLxx`) type across f, and ν is not eroded.

2. Otherwise, the face is *free*.

In an **adaptive** case:

1. Let f be the face of a continuum element e. Let ν be the neighbor (if any) of e across f and let π be the pseudo-neighbor, or p-neighbor for brevity, (if any) of e across f. In adaptivity the p-neighbor is the currently active (no children) lower-level (bigger) element located across an element's face, and results from the fact that the current element has been produced by a localized mesh refinement. Recall that: *a*) neighbor and p-neighbor are mutually exclusive, i.e. it may not be $\nu > 0$ and $\pi > 0$ at the same time; and *b*) a p-neighbor has no children by definition.

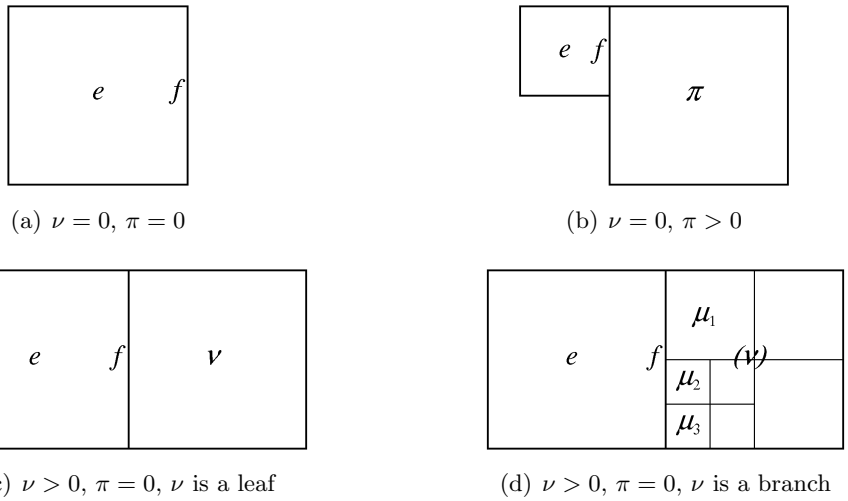
2. If $(\nu = 0)$ and $(\pi = 0)$, see Figure 2(a), then f is free.

3. Else if $(\nu = 0)$ and $(\pi > 0)$, see Figure 2(b), then:

   - If π is eroded, then f is free.
   - Otherwise f is not free.

4. Otherwise $(\nu > 0)$ and $(\pi = 0)$, see Figure 2(c) and 2(d). Then:

   - If ν has no children (is a *leaf* in the adaptivity tree), see Figure 2(c), then:
     - If ν is eroded, then f is free.
     - Otherwise f is not free.
   - Else ν has children (is a *branch* in the adaptivity tree), see Figure 2(d). We search all the *currently active* descendents μ_i of ν (at any level of the adaptivity tree). A descendent is currently active if it is a leaf, i.e. if it has no children of its own. In the example of Figure 2(d), these are μ_1, μ_2 and μ_3. Then:
     - If all μ_i are eroded, then f is free.
     - Otherwise f is not free.



(a) $\nu = 0$, $\pi = 0$

(b) $\nu = 0$, $\pi > 0$

(c) $\nu > 0$, $\pi = 0$, ν is a leaf

(d) $\nu > 0$, $\pi = 0$, ν is a branch

Figure 2: Free faces of continuum elements in adaptivity.

2.3 Contact-driven adaptivity (new model)

In this Section we propose a new model of contact-driven mesh adaptivity (alternative to the point cloud-based model shorty mentioned in a previous Section). For the moment, the model is operational only in the sequential (not parallel) version of the code. The proposed algorithm is roughly similar to the algorithm for FSI-driven adaptivity detailed in reference [32].

In adaptive calculations with contact-impact, mesh refinement and un-refinement should occur in the vicinity of contacting surfaces (in order to enhance the resolution of contact detection and the accuracy of contact force calculations), in addition (or in alternative) to the wavefronts which are tracked via `WAVE` or `INDI` directives. Since this (contact-based) type of mesh adaptation is independent from `WAVE` or `INDI` directives, but is related to the chosen contact model, it seems preferable to embed the corresponding input directive within the contact directives themselves (e.g. `PINB` or `GPIN`).

In principle, the constraints on mesh size resulting from contact should have to be merged with any constraints due to `WAVE`, `INDI` or any other (still to be developed) adaptivity directives. The minimum local mesh size resulting from all such constraints would then have to be retained. However, at the moment the implementation of `WAVE` and `INDI` are such that these models may not be combined in the same calculation. For simplicity, also the present adaptive contact model is initially developed and tested as an independent model, incompatible with `WAVE` or `INDI` (however, it should be compatible with threshold-driven adaptivity `THRS`). Once all these models are well tested separately, they will have to be re-implemented in a compatible way by developing a suitable combination strategy.

2.3.1 New syntax of the `PINB` directive

The `PINB` contact directive has the following syntax (see [1] for details):

```
PINB < PENA <SFAC sfac> >
     ( $ BODY ; SELF $
       < $ DMIN dmin ; MLEV mlev $ >
       < DIAM diam >
       < HARD hard >
       /LECT/ )
     < EXCL (PAIR n1 n2) >
     < ADAP LMAX lmax <SCAL scal> <SCAS scas> <NOUN> >
```

The new `ADAP` sub-directive (in red above) introduces adaptivity-related data for the pinball-based contact model. At the moment, the only parameters that can be given are: `LMAX` (denoted $L_{\max}$ in the following), the desired maximum mesh refinement level near the contacting surfaces; `SCAL` (denoted ϕ or ϕ_n), an optional scaling factor applied to non self-contacting bodies and used to govern the extent of the adapted zone; and `SCAS` (denoted ϕ_s), a similar factor for self-contacting bodies. The scaling factors are discussed in detail in one of the following Sections. By default $\phi_n = 1.0$ and $\phi_s = 0.55$. The

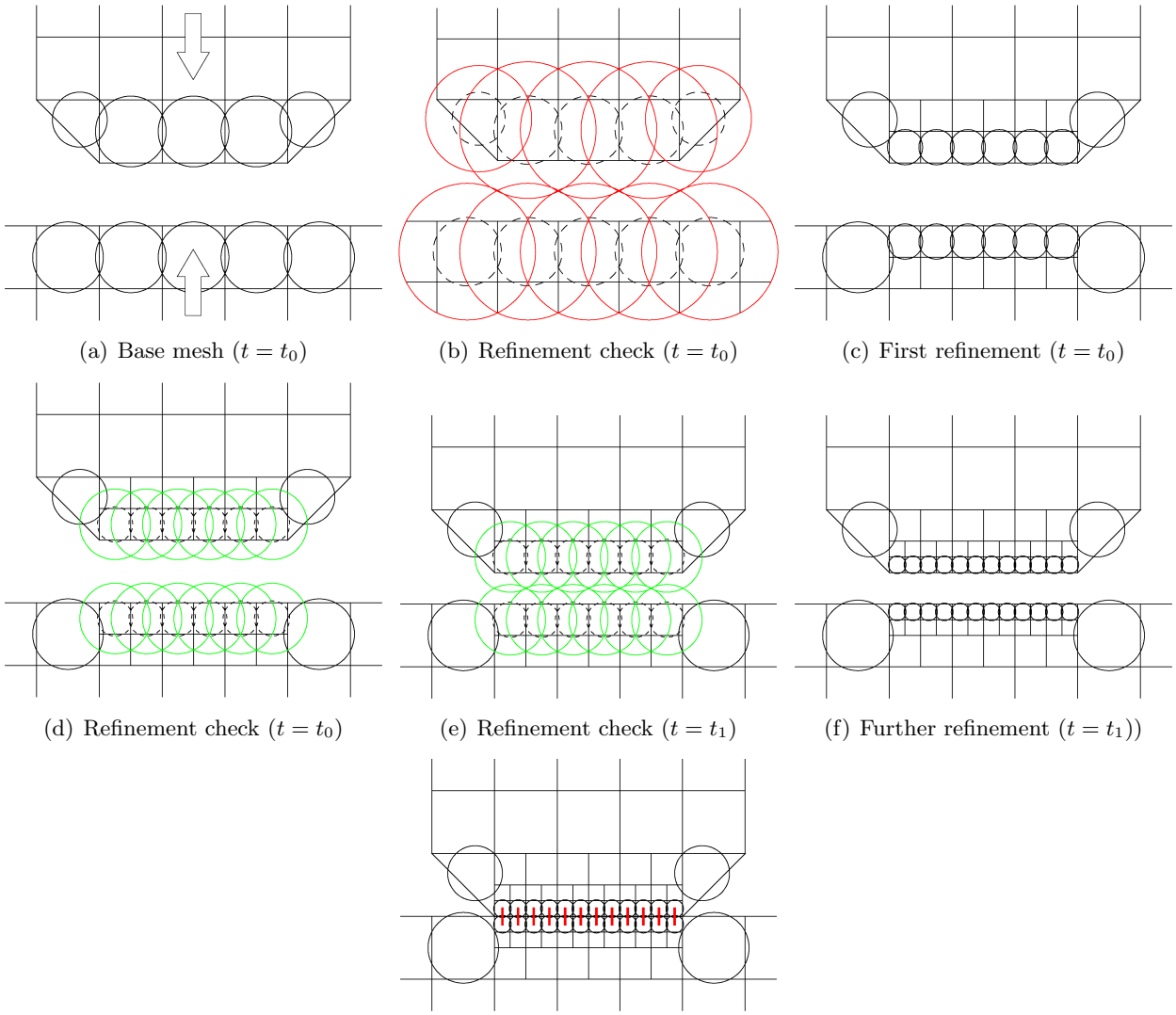
optional keyword `NOUN` can be specified in order to disable mesh unrefinement due to contact-driven adaptivity.

2.3.2 Choosing the maximum adaptive refinement level

In analogy with FSI-driven adaptivity, the practical convention is adopted that the user always specifies the `PINB` data referred to the *base* mesh, also in the case of an adaptive calculation. Then, if contact-driven adaptivity is needed, the `ADAP` keyword is added to the `PINB` directive and the `LMAX` keyword is used to introduce the desired maximum refinement level (`lmax`) of the mesh near the contacting surfaces. In this way, various levels of contact-induced refinement can be tried out by changing only the `lmax` value.

2.3.3 Adapting the mesh

A conceptual example of contact-driven mesh refinement is presented in Figure 3.



(a) Base mesh ($t = t_0$) (b) Refinement check ($t = t_0$) (c) First refinement ($t = t_0$)

(d) Refinement check ($t = t_0$) (e) Refinement check ($t = t_1$) (f) Further refinement ($t = t_1$))

(g) Contacts detected ($t = t_2$)

Figure 3: Example of contact-driven mesh refinement.

Consider two solid bodies moving towards each other, as indicated by the arrows in Figure 3(a). The bodies are meshed rather coarsely. In order to model the contact, (parent) pinballs are embedded in each body. Assume (for simplicity) that no element failure (nor erosion) has to be modelled, so that it is sufficient to embed pinballs only in the superficial elements of the two bodies, as shown by the black circles in 3(a).

Refining the mesh near the contacting surfaces of the two bodies brings various benefits. First, the accuracy (resolution) of contact detection is improved. Since the pinballs extend slightly outside the corresponding elements, contact has a tendency to be detected "too early", when the real surfaces aren't touching each other yet. Therefore, using smaller pinballs reduces this type of inaccuracy. Second, after contact has started the contact surface may deform in a complex way, so that using smaller elements increases the accuracy of the solution.

However, it is also clear that in general there is no need to have too fine a mesh in regions of the bodies far from the contact zones, unless one wants to follow some sharp gradients travelling inside the bodies themselves. So it is desirable that, when contact terminates, i.e. when the two bodies start rebounding, the local mesh be progressively unrefined so as to speed up the calculation.

One essential aspect of this type of refinement is that the mesh must be refined *before* contact first starts, i.e. when there is still a certain gap between the bodies. Refining when contact has been already detected (with a coarse mesh) would be "too late" and would spoil the accuracy of the model.

The refinement is therefore based on the distance between the two candidate contacting surfaces. As shown in Figure 3(b), one simple way to check for refinement is to use some fake "magnified" pinballs, shown as red circles in 3(b), and to check the interpenetrations between them as if they were the real pinballs. To each parent (0-level) pinball of the base mesh (black dotted circle) we associate a magnified pinball (red circle) with the same center. By default, a magnification factor of 2.0 is taken (since the adaptive mesh refinement and un-refinement algorithm is based on powers of two), but the value may be changed by means of the `SCAL` keyword (ϕ coefficient). Then, interpenetrations are checked and the elements associated with inter-penetrating (red) pinballs are refined (once). In doing so, the algorithm presented in the previous Sections ensures that the created children elements will also be endowed with (smaller) pinballs, as shown in 3(c). For simplicity of drawing, and assuming that no element erosion is activated, only the "external" children elements would actually receive pinballs (a strategy for ensuring this behaviour must be devised).

Next, a further check takes place, now involving (only) pinballs of the next level (level 1), as shown in Figure 3(d). Again, each one of these pinballs is replaced by a magnified fake pinball (now in green). No interpenetration is detected, and so no further refinement takes place at this step. *A fortiori*, no contact is detected between the *real* (not magnified) pinballs at this step, and the calculation goes on.

After some time steps ($t = t_1$), the gap between the two bodies has narrowed and some interpenetrations between the (green) fake (magnified) pinballs is detected, as shown in 3(e). Then, the corresponding elements are (futher) refined, see 3(f). Let us assume for simplicity that, with this second refinement, the maximum prescribed refinement level has been reached, so that no further refinement can take place.

In the configuration shown in 3(f) no contacts between real pinballs are detected, so the calculation goes on. Finally, after some more time ($t = t_2$) the gap between the two bodies has become so small that some real contacts are detected, as shown in Figure 3(g).

At this moment the pinballs have become so small that the inaccuracy of penetration detection is negligible. Furthermore, the fine discretization (small elements) along the contacting surfaces ensures high accuracy of contact force calculations.

The actual refinement (and unrefinement) procedures, inspired by the simple example considered above, will now be detailed. In the following, the (parent) pinball associated with an element i is indicated by p_i, while the corresponding *magnified* pinball is indicated by π_i (where π_i is ϕ times bigger than p_i).

Two loops are performed at the beginning of each time step to adapt the mesh parts potentially subjected to contact. In the first loop, "coarse" contact-candidate elements which now find themselves "near" some other contact-candidate elements are (progressively) *refined*, while in the second loop "fine" contact-candidate elements which now find themselves "far" from any other contact-candidate elements are (progressively) *unrefined*.

Each loop proceeds by examining *one level at a time*. The refinement loop does this in *increasing* level order (from level 0 to level $L_{\max} - 1$), while the unrefinement loop does this in *decreasing* level order (from level $L_{\max} - 1$ to level 0). Note that elements at level $L_{\max}$ are never examined directly. In fact, they need no refinement since they are already at the maximum refinement level. Nor are

they (directly) examined during unrefinement, because the unrefinement process is nominally applied to the *parent* element and not to its children.

2.3.4 Mesh refinement loop

The following **refinement criterion** is adopted. An *active* element a at level L is refined (once) if it has an associated (parent) pinball p_a which penetrates another pinball p_b not belonging the same body (unless self-contact has been prescribed), associated with another active element b of level $M \geq L$. Recall that active elements in adaptivity are those which are currently used but have no children, i.e. they are *leaves* in the elements tree.

Therefore, starting at level 0, all active elements at level 0 (i.e. all active *base* elements) which satisfy the refinement criterion are refined to level 1. Next, we examine active elements at level 1. If such an element satisfies the refinement criterion, then it is refined to level 2. And so on, until we examine elements in level $L_{\max} - 1$. The proposed algorithm (provisional) is shown in Algorithm 1.

Algorithm 1 Provisional algorithm for contact-driven mesh refinement.

1. Initialize the level counter: $L = -1$.
2. Increment the level counter: $L = L + 1$.
3. Loop over the elements i.
4. If element i has no (parent) pinball p_i, cycle.
5. If element i is unused, cycle.
6. If element i has a level $L_i \neq L$, cycle.
7. If element i is inactive, i.e. if it has children, cycle.
8. If the magnified pinball π_i of element i does *not* penetrate the magnified pinball π_j of any other element j such that:
   - j has a (parent) pinball p_j,
   - j is used,
   - j has level $L_j \geq L$,
   - j is active (i.e., it has no children),

   then cycle.
9. Refine element i.
10. End loop over the elements.
11. If $L < L_{\max} - 1$, `GO TO` 2.
12. End of refinement.

It is clear that immediate unrefinement of an element following a refinement of the same element performed during the *same* time step (so-called *ping-pong* effects) should be avoided. To this end, we state the following rule, which is quite general and should be satisfied by any mesh adaptation algorithm.

Rule (ping-pong). An unrefinement algorithm should *not immediately* undo what a refinement algorithm has just done.

2.3.5 Mesh unrefinement loop

The following **unrefinement criterion** is adopted. An *inactive* element i at level L is unrefined (once) if all its *children* are *active* (i.e. they have no children of their own) and i has an associated (parent) pinball p_i which *does not* penetrate *any* other pinball p_j not belonging to the same body (unless self-contact has been prescribed), associated with another *active* element j of level $M \geq L$.

Therefore, starting at level $L_{\max} - 1$, all inactive elements at this level satisfying the unrefinement criterion are unrefined once to level $L_{\max} - 2$, and in doing so they become active while their children become unused. At this particular level there would be no need to check that the children are active, since they are at the maximum level $L_{\max}$. Next, we examine inactive elements at level $L_{\max} - 2$. If such an element has all active children and satisfies the unrefinement criterion, then it is unrefined to level $L_{\max} - 3$. And so on, until we examine elements at level 0, i.e. the base elements. The proposed algorithm (provisional) is shown in Algorithm 2.

Algorithm 2 Provisional algorithm for contact-driven mesh unrefinement.

1. Initialize the level counter: $L = L_{\max}$.
2. Decrement the level counter: $L = L - 1$.
3. Loop over the elements i.
4. If element i has no (parent) pinball p_i, cycle.
5. If element i is unused, cycle.
6. If element i has a level $L_i \neq L$, cycle.
7. If element i is active, i.e. if it has no children, cycle.
8. If any of the children c of element i is inactive, i.e. if c has its own children, cycle.
9. If the magnified pinball π_i of element i penetrates the magnified pinball π_j of at least another element j such that:
   - j has a (parent) pinball p_j,
   - j is used,
   - j has level $L_j \geq L$,
   - j is active (i.e., it has no children),

   then cycle.
10. Unrefine element i.
11. End loop over the elements.
12. If $L > 0$, `GO TO` 2.
13. End of unrefinement.

2.3.6 Dealing with pinball size options and with numbering indepence

The provisional versions of the refinement and unrefinement algorithms sketched in the previous Section will now be finalized. First, the algorithms must be rendered compatible with the various options existing in the code, which may affect the size of the pinballs used to detect the contact. Second, one must ensure that the algorithms are numbering-independent, i.e. that results do not depend upon the particular numbering of the elements (and thus of the pinballs) in the mesh.

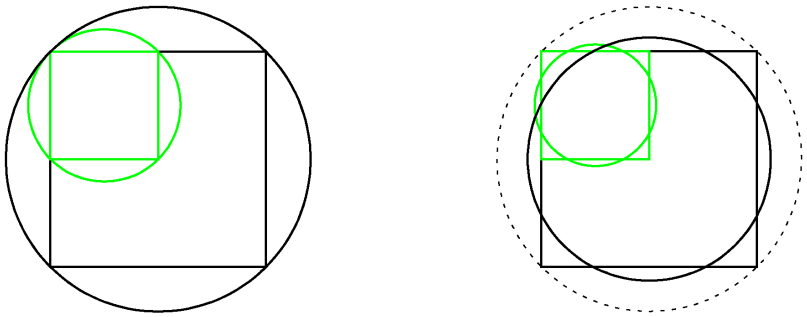
In EPX various input options can be used to control the size of pinballs used to detect the contact. For a parent (0-level) pinball, the center is always computed as the centroid of the element nodes, i.e. as the arithmetic average of the element's nodal coordinates.

By default (no option specified) or by specifying the `NEQV` option (`OPTI PINS NEQV`), the parent pinball diameter is computed so as to exactly contain all the nodes of the element (*encompassing* pinball), as shown by the black circle in Figure 4(a). Note that in this Figure we consider a regular (square) element for simplicity, but the rule holds for any shape of the element. As the element is refined, its children have size one half of the parent, see the green square for example. The same happens to children pinballs (green circle).

By specifyng the `EQVL`, `EQVD` or `EQVF` options, the pinball size is computed such that the pinball's volume equals the volume of the element (*equivalent* pinball), thus resulting in a smaller pinball, see the black solid circle in Figure 4(b), than the default (dotted circle). The difference between `EQVL`, `EQVD` and `EQVF` is that the first one applies to parent (0-level) pinballs only, the second one applies to descendent pinballs (at any level) which are generated if a hierarchic pinball method is activated, and the third one applies ony to descendents at the *final* (i.e. highest) level of the hierarchy (see [1] for details).

The reason for using an equivalent as opposed to an encompassing pinball is that in most cases the former is a better approximation of the real element shape, thus leading to slightly more accurate contact detection. Note in fact that equivalent pinballs "protrude" outside the element boundaries less than encompassing ones. Therefore, contact detection occurs at a smaller (spurious) gap between real elements. However, this introduces the risk of overlooking some contacts near the element corners, which are not contained within the pinball as shown in 4(b).

Although the default in EPX is to use encompassing pinballs, according to the users' manual in most applications it is preferable to use equivalent pinballs at the highest level, that is option `EQVF` if a hierarchic pinball method is chosen, or option `EQVL` if a basic pinball method is chosen.



(a) Encomapssing pinball (b) Equivalent pinball

Figure 4: Options affecting the size (volume) of pinballs.

Returning now to contact-driven adaptivity algorithms, one sees that the use of encompassing fake (magnified) pinballs for the detection of elements (and pinballs) to be refined is probably a better choice than equivalent pinballs. In fact, note (Figure 4(a)) that in this case all children pinballs (green circles) lie *inside* the parent pinball (black circle), and this at any level of the hierarchy. This ensures "safer" (more conservative) detection of the mesh zones (elements and associated pinballs) to be refined in view of possible contact.

In principle, the above property could be exploited in order to simplify the algorithms. In fact, it would allow to restrict the number of p_j pinballs to be checked for penetration of the p_i pinball at step 8 of the refinement algorithm and at step 9 of the unrefinement algorihm. The condition $L_j \geq L_i$ could be replaced by $L_j = L_i$ (i.e., considering only pinballs at the same level), but provided one would check *inactive* pinballs (inactive elements) in addition to active ones, i.e. provided one at the same time would remove the condition "j is active". However, since the internal status of an inactive pinball cannot be guaranteed to be consistent (such pinballs are not updated at every step), it is preferred to leave this part of the algorithm as in the provisional version.

For the sake of simplicity, we retain that the fake pinballs to be used for contact-driven adaptivity be constructed by magnifying the corresponding real (parent) pinballs, *whatever their type* (encompassing or equivalent). Note incidentally that, even in the case of a hierarchic pinball method, only parent pinballs, at each level of adaptive mesh refinement, are used to construct the fake ones. Thus, the chosen type of parent pinball will be retained also for the fake pinballs.

If necessary, any negative effect of using equivalent (i.e., smaller) parent pinballs and thus equivalent fake pinballs can be easily compensated by choosing a scaling factor (`SCAL` keyword of the `PINB` directive, see above) greater than 1.0 (the default value).

Next, we examine the independence of the algorithmic results from the actual numbering order chosen for the elements (and the pinballs) in the mesh. This property is crucial for an industrial use of the code and *must* be satisfied.

The provisional algorithms sketched above are directly derived from the similar ones of reference [32], conceived for FSI-driven adaptivity. However, while FSI algorithms deal with one element at a time, here we deal with *couples* of elements because contact (or, more precisely, interpenetration) checks are symmetric: if pinball p_i penetrates pinball p_j, then p_j penetrates p_i (and reciprocally). This implies that at point 9 of the refinement algorithm one has to refine element (pinball) j in addition to element i.

Furthermore, a pinball p_i can penetrate *more than one* other pinball at the same time (say p_{j_1}, p_{j_2} etc.). Therefore, in order to be sure to achieve mesh-numbering independence it seems necessary to check p_i for interpenetration with *every* other p_j satisfying the conditions stated above, before possibly refining p_i and all the penetrated p_{jk}. Hence, the algorithms must be modified so that any refinement or unrefinement operations are performed *at the end* of each pass over a level L, rather than immediately as in the provisional versions. The complete versions of the algorithms (now including also consideration of element erosion and contact "bodies", also in the case of self-contact, but not yet optimized) are given below in Algorithms 3 and 4. By convention, if an element i belongs to a self-contacting body B_i, its body index is set to $-B_i$.

Algorithm 3 Non-optimized algorithm for contact-driven mesh refinement.

1. Initialize the level counter: $L = -1$.
2. Increment the level counter: $L = L + 1$.
3. Wipe out the list of elements to be refined.
4. Loop over the (parent) pinballs p_i. Let i be the element and B_i be the body index associated with p_i.
   (a) If element i has been eroded, cycle on i.
   (b) If element i is unused, cycle on i.
   (c) If element i has a level $L_i \neq L$, cycle on i.
   (d) If element i is inactive, i.e. if it has children, cycle on i.
   (e) Loop over the (parent) pinballs p_j with $p_j > p_i$. Let j be the element and B_j be the body index associated with p_j.
      i. If element j has been eroded, cycle on j.
      ii. If element j is already in the list of elements to be refined, cycle on j.
      iii. If element j is unused, cycle on j.
      iv. If element j has a level $L_j < L$, cycle on j.
      v. If element j is inactive, i.e. if it has children, cycle on j.
      vi. If p_j belongs to the same body as p_i (i.e. if $B_j = B_i$) *and* self-contact (`SELF`) has *not* been activated on B_i (i.e., if $B_i > 0$), then cycle on j.
      vii. If the magnified pinball π_i penetrates the magnified pinball π_j, then add both element i and element j to the list of elements to be refined and **continue** loop on j. Note that element i might already be in the list.
   (f) End loop over the elements j.
5. End loop over the elements i.
6. Refine all the elements in the list.
7. If $L < L_{\max} - 1$, `GO TO` 2.
8. End of refinement.

2.3.7 Optimized algorithms using fast search

The mesh-adaptation algorithms presented above in Algorithms 3 and 4 are not directly suitable for use in large applications because they are not optimized. Therefore, such algorithms have not been implemenetd in the code.

Fast search of contacting (magnified) pinballs must be performed to achieve reasonable computing times. Using a fast search algorithm as opposed to the trivial, brute-force strategy that would consist in simply checking each pinball for penetration with *any other* pinball, has some implications in the form of the algorithm itself.

The fast search strategy adopted is based upon a *search by box* procedure, see [40]. The space is subdivided into hexagonal boxes (rectangular parallelepipeds) oriented along the global axes. Then, a list of the items (i.e., the magnified pinball centers) to be checked contained inside each box is built up. Finally, the search for inter-penetration between couples of magnified pinballs is performed by neighboring boxes. That is, each magnified pinball is checked for interpenetration not with all other magnified pinballs, but only with those contained either in the same box, or in a directly neighboring box. For each box, there are (up to) 8 neighboring boxes in 2D, or 26 neighboring boxes in 3D. This greatly reduces the number of interpenetration checks to be performed and so speeds up the search.

For optimum performance, the size of the boxes must be as small as possible, but not too small so that some interpenetrations could be overlooked. The chosen size of the box is taken equal to the maximum diameter of the magnified pinballs to be checked. To achieve element refinement, we proceed as shown in Algorithm 5 (this is the version actually implemented in EPX).

In order to illustrate the optimized mesh unrefinement algorithm, we introduce first the notion of **unrefinable pinball**.

Definition. An *unrefinable pinball* p_i is a (parent) pinball associated with an element i which could be unrefined, i.e. which is currently used but inactive in the adaptivity tree, and whose children are

Algorithm 4 Non-optimized algorithm for contact-driven mesh unrefinement.

1. Initialize the level counter: $L = L_{\max}$.
2. Decrement the level counter: $L = L - 1$.
3. Wipe out the list of elements to be unrefined.
4. Loop over the (parent) pinballs p_i. Let i be the element and B_i be the body index associated with p_i.
   (a) If element i has been eroded, cycle on i.
   (b) If element i is unused, cycle on i.
   (c) If element i has a level $L_i \neq L$, cycle on i.
   (d) If element i is active, i.e. if it has no children, cycle on i.
   (e) If any of the children c of element i is inactive, i.e. if c has its own children, cycle on i.
   (f) Provisionally add element i to the list of elements to be unrefined.
   (g) Loop over the (parent) pinballs p_j with $p_j > p_i$. Let j be the element and B_j be the body index associated with p_j.
      i. If element j has been eroded, cycle on j.
      ii. If element j is already in the list of elements to be unrefined, cycle on j.
      iii. If element j is unused, cycle on j.
      iv. If element j has a level $L_j < L$, cycle on j.
      v. If element j is inactive, i.e. if it has children, cycle on j.
      vi. If p_j belongs to the same body as p_i (i.e. if $B_j = B_i$) *and* self-contact (`SELF`) has *not* been activated on B_i (i.e., if $B_i > 0$), then cycle on j.
      vii. If the magnified pinball π_i penetrates the magnified pinball π_j, then remove i from the list of elements to be unrefined and **exit** loop on j.
   (h) End loop over the elements j.
5. End loop over the elements i.
6. Unrefine all the elements in the list.
7. If $L > 0$, `GO TO` 2.
8. End of unrefinement.

all currently active.

Based upon this definition, and by directly exploiting the *no ping-pong* rule fitsy introduced in a previous Section, the **unrefinement criterion** can be reformulated as follows. An *unrefinable pinball* p_i associated with an element i at level L is unrefined (once) if, by *pretending* that i would be active (i.e. with no children), p_i would *not* be refined by the previously defined refinement algorithm.

To achieve element unrefinement, we proceed as shown in Algorithm 6 (this is the version actually implemented in EPX).

2.3.8 Treatment of self-contact

In case of self-contact, straightforward application of the algorithms detailed above with default values of refinement parameters often produces a *completely refined* mesh. That is, the mesh of the contacting body declared by the `SELF` keyword in the pinball directive is immediately and uniformly refined to the maximum chosen level `LMAX`.

This is illustrated in Figure 5. Consider a simple patch of (regular) quadrilateral elements (`square`), declared as a self-contacting body (`SELF`) with adaptive refinement up to `MLEV` 3 upon contact:

```
LINK COUP SPLT NONE
     PINB SELF MLEV 0 LECT square TERM
          ADAP LMAX 3 SCAL 1.0
```

Then, if the patch contains only 2×2 elements like in 5(a) we obtain the expected result: at time 0 no refinement is performed. However, if we use a larger patch of, say, 3×3 elements, then an unexpected behaviour appears. In 5(b) we see the *base* mesh for this case, i.e. without activating the

Algorithm 5 Optimized algorithm for contact-driven mesh refinement.

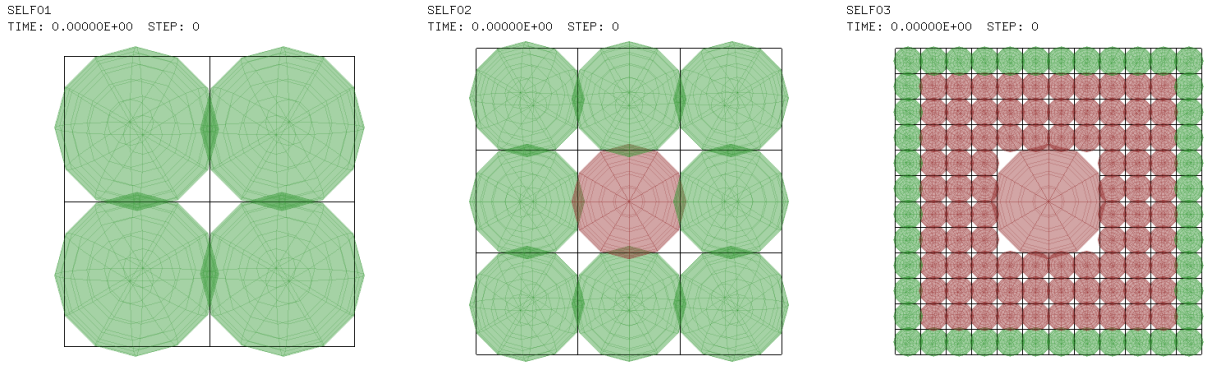
1. Initialize the level counter: $L = -1$.
2. Increment the level counter: $L = L + 1$.
3. Wipe out the list R of pinballs to be refined.
4. Prepare a list C of *candidate* magnified pinballs *for refinement*. These are all the (parent) magnified pinballs π_i satisfying *all* following conditions (where i is the element to which π_i is associated):
   (a) π_i is currently active.
   (b) i is not eroded.
   (c) i is at a level L_i of the adaptivity tree equal to or higher than L ($L_i \geq L$). Note that this also ensures that i is used in the adaptivity, since unused elements in adaptivity have $L_i = 0$. (Note, however, that i could be inactive in adaptivity, hence we need the next test.)
   (d) i has no children, i.e. it is active (a leaf) in adaptivity.

   The list is stored in the array `ACTIVE_PINBALLS`. Note that further checks concerning the pinballs body, also accounting for self-contact if necessary, are performed as part of the inter-penetration calculations below, see Step 6(e) of this algorithm.
5. Build up the box-based fast search data structure for the candidate (magnified) pinballs.
6. Execute the fast search algorithm. For each box:
   (a) Prepare a list N of all magnified pinballs π_i from C contained (i.e. whose centroid lies) either in the current box or in one of the direct neighbor boxes.
   (b) Loop on all π_I in the current box only (*not* in the neighbor ones).
   (c) Loop on all π_J in the list N, such that $J > I$. (Note that here we use capital indexes to indicate that I and J are the indexes of the pinballs and *not* the indexes of the associated elements.)
   (d) Let B_I and B_J denote the bodies to which the two pinballs belong.
   (e) If $B_I < 0$ and $B_J = B_I$ (i.e. the two pinballs belong to the same self-contacting body), or $B_I \neq B_J$ (i.e. the two pinballs belong to different bodies), then check for penetration: if penetration is detected, add both pinball p_I and pinball p_J to the list R of pinballs to be refined (note that they could be already in the list).
7. Destroy the box-based fast search data structure for the candidate (magnified) pinballs.
8. Refine (once) all the elements i associated with a pinball p_i in the list R, provided i is not yet at the maximum refinement level (i.e. if $L_i < L_{\max}$).
9. If $L < L_{\max} - 1$, `GO TO` 2.
10. End of refinement.

Algorithm 6 Optimized algorithm for contact-driven mesh unrefinement.

1. Initialize the level counter: $L = L_{\max}$.
2. Decrement the level counter: $L = L - 1$.
3. Wipe out the (fake) list R of pinballs to be refined.
4. Build up a list U of unrefinable pinballs at level L.
5. Prepare a list C of candidate pinballs for (fake) refinement exactly like in the refinement loop (`ACTIVE_PINBALLS`). This corresponds to Step 4 of Algorithm 5
6. Add to this list all the unrefinable pinballs: $C = C \cup U$. Note that these are guaranteed *not* to be already on the list, since they are currently inactive. Sort the list C in ascending order (just for clarity).
7. Search all pinballs to be refined and put them in a (fake) list R. This corresponds to Steps 5–7 of Algorithm 5
8. Unrefine all the elements i associated with an unrefinable pinball p_i which would *not* be refined. That is, p_i is in the list U but it is *not* in the (fake) refinement list R.
9. If $L > 0$, `GO TO` 2.
10. End of unrefinement.

`PINB` ... `ADAP` keyword. The picture in 5(c) shows what happens by activating adaptivity. The entire mesh, with the notable exception of the central element, is immediately refined to the maximum level.



(a) SELF01 (b) SELF02 (c) SELF03

Figure 5: Spurious mesh refinement (*chain reaction*) in the self-contact case.

The reason for this apparently weird behaviour is explained in Figure 6. Consider for simplicity a patch of three *regular* square elements A, B and C in 2D. The encompassing pinballs p_A, p_B and p_C are shown as black circles, while the corresponding *magnified* pinballs π_A, π_B and π_C are shown as green circles. The *default* magnification factor $\mu_0 = 2.0$ has been assumed, i.e. the radius of the green pinballs is twice the radius of the black pinballs.

One sees that the magnified pinball π_A penetrates *both* π_B and π_C. Now, the interpenetration between π_A and π_B does *not* induce any refinement, because the geometrical checking algorithm considers as *invalid* (and therefore skips) the penetration between two pinballs belonging to elements which *share a common face*, like in the case of A and B. However, the penetration between π_A and π_C is valid, since elements A and C do not have any common face, so that both elements are refined.

It is now clear why a row of just two elements is not spuriously refined, as shown in Figure 5(a) while in a row of three elements the outer elements are refined, but not the central one, as shown in Figure 5(c). With 4 or more elements in a row, a sort of *chain rection* is triggered, which causes refinement of *all* elements, and this up to the maximum allowed refinement level.

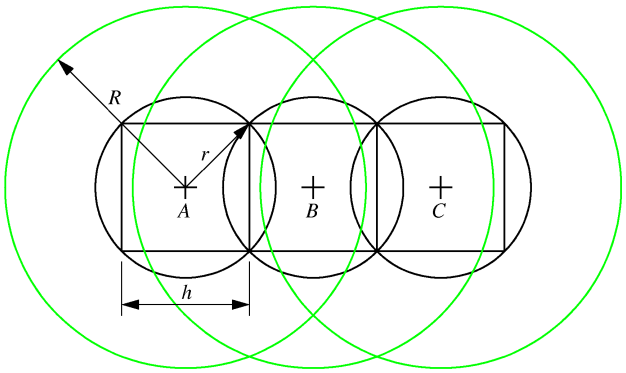


Figure 6: Illustration of the chain reaction phenomenon in the self-contact case.

In order to avoid a chain reaction the only possibility seems to use a smaller value of the magnification factor μ. In fact, any *ad-hoc* tests trying, for example, to establish whether elements A and C are second-level contiguous (they are both contiguous to a common element B) would be extremely expensive and, more importantly, would completely spoil the purpose of the self-contact algorithm. In fact, elements A and B *could* eventually come into (self) contact, either due to extreme deformation of the body or, perhaps more likely, due to failure and erosion of element B.

Note, incidentally, that activation of self contact is in principle necessary for any body which can undergo severe failure and element erosion, if one wants to avoid non-physical inter-penetration of

the *macroscopic fragments* (i.e. patches of not yet eroded elements) produced by the element erosion process.

The (upper) limit of the magnification factor μ can be easily computed in an ideal situation (regular elements) such as that of Figure 6. Let h be the mesh size, i.e. the side of the square elements. Then, the radius r of the *encompassing* pinballs is:

$$r = \sqrt{2}\,\frac{h}{2} = \frac{h}{\sqrt{2}} \tag{1}$$

The radius R of the magnified pinballs is:

$$R = \mu\, r \tag{2}$$

where μ is the magnification factor ($\mu = R/r$ by definition). The maximum value of R in order to avoid the chain reaction is clearly given by the distance between the centres of two elements:

$$R_{\max} = h \tag{3}$$

Therefore, one obtains for the maximum magnification factor:

$$\mu_{\max} = \frac{R_{\max}}{r} = h\,\frac{\sqrt{2}}{h} = \sqrt{2} = 1.414 \tag{4}$$

The values of the maximum magnification factor in other situations, such as when using *equivalent* instead of encompassing pinballs, or in a regular 3D mesh of cubes, can be obtained similarly and are summarized in Table 1.

Case	Encompassing pinballs	Equivalent pinballs
2D continuum (squares)	$\sqrt{2} = 1.414$	$\sqrt{\pi} = 1.772$
3D continuum (cubes)	$\frac{2}{\sqrt{3}} = 1.154$	$\sqrt[3]{\frac{4\pi}{3}} = 1.611$

Table 1: Maximum magnification factor $\mu_{\max}$ in 2D and 3D regular continuum meshes.

As shown in the Table, in all cases $\mu_{\max}$ is less than the value 2.0 which had been assumed so far by default. Smaller values of $\mu_{\max}$ hold in 3D with respect to 2D. Also, the values of $\mu_{\max}$ for encompassing pinballs are smaller than those for equivalent pinballs, as it could be expected. This shows the interest of using equivalent pinballs whenever possible, especially in 3D calculations.

In order to avoid the chain reaction problem in practical applications, various alternative strategies could be set up. For example:

a) Use the optional `SCAL` keyword (ϕ coefficient) to scale down the default magnified pinballs so that the actual magnification factor falls below $\mu_{\max}$.

b) Allow the user to override the (default) magnification factor ($\mu_0 = 2.0$) by means of a (new) *ad-hoc* optional keyword.

c) Treat the problem internally by automatically using a smaller magnification factor than the default for all pinballs which have the self-contact property enabled (i.e. whose body has a negative index).

Each strategy has it own advantages and drawbacks. The first one is also the simplest, but it leaves responsibility entirely to the code user. However, this may not be a serious drawback, since in any case it should be kept in mind that the above values of $\mu_{\max}$ are valid only in the *ideal* case of regular continuum elements. If the elements are irregular or get distorted, or in case one uses other element shapes (e.g. triangles, tehtrahedra) or other element classes (e.g. shells), the limiting value is much harder (or impossible) to determine exactly, so in any case the user would have to choose it. If too large a magnification factor is chosen, the mesh will be excessively refined, possibly also leading to a chain reaction.

According to strategy *a)*, the value of the `SCAL` parameter (ϕ factor) to be chosen in order to avoid chain reaction is given by:

$$\phi_{\max} = \frac{\mu_{\max}}{\mu_0} \tag{5}$$

where μ_0 is the default magnification factor ($\mu_0 = 2.0$). The values of $\phi_{\max}$ for the various ideal cases are shown in Table 2.

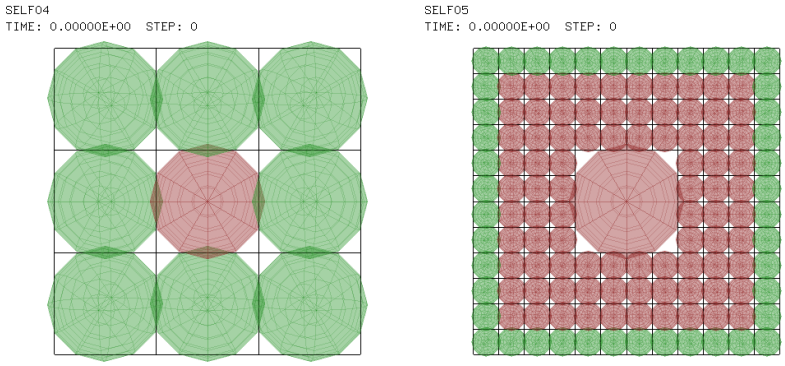
Case	Encompassing pinballs	Equivalent pinballs
2D continuum (squares)	$\frac{\sqrt{2}}{2} = 0.707$	$\frac{\sqrt{\pi}}{2} = 0.886$
3D continuum (cubes)	$\frac{1}{\sqrt{3}} = 0.577$	$\sqrt[3]{\frac{\pi}{6}} = 0.806$

Table 2: Maximum scaling factor $\phi_{\max}$ for self-contact in 2D and 3D regular continuum meshes.

In order to check the above values, the test of Figure 5(b) is repeated by adding a "safe" value of the ϕ coefficient (e.g., $\phi = 0.880$ in this particular case) by means of the `SCAL` keyword:

```
LINK COUP SPLT NONE
     PINB SELF MLEV 0 LECT square TERM
          ADAP LMAX 3 SCAL 0.880
```

This avoids the chain reaction, as shown in Figure 7(a). A value of $\phi = 0.890$, instead, triggers the reaction, see 7(b).



(a) SELF04, $\phi = 0.880$ (b) SELF05, $\phi = 0.890$

Figure 7: Effects of the choice of the scaling factor ϕ in the self-contact case.

2.3.9 Choice of the scaling factor

By summarizing the above findings, one might state the following practical guidelines for the choice of the scaling factor ϕ in contact-driven adaptivity models.

If the model **has no self-contacting body** (no `PINB SELF` is present), then the choice of the scaling factor ϕ (`SCAL` keyword) is free. By default the code will use $\phi = \phi_0 = 1.0$ which, combined with the (default) pinball magnification factor $\mu_0 = 2.0$ produces magnified pinballs twice the size of real ones. The extent of the refined zone can be adjusted by the user, if needed, by choosing a value of ϕ greater than one (to enlarge the refined zone) or smaller than one (to reduce it). Enlarging the refinement zone usually produces a more smoothly-graded refined mesh, but it also increases the number of elements to be computed.

If the model **has at least one self-contacting body**, then leaving the ϕ factor to its default value of 1.0 will most probably induce a uniform refinement (chain reaction) in the self-contacting body (or bodies), although not in ordinary bodies. This is usually not acceptable. The user may avoid chain reaction by prescribing a scaling factor $\phi < 1.0$. The values in Table 2 are indications of maximum values of ϕ for various *ideal* cases. In real cases, the value to be chosen will probably

have to be smaller (in order to account for element deformation), and some experimentation might be required.

An obvious pitfall of the above strategy is that, in cases with self-contact, the use of a single (*global*) value of ϕ for all the bodies is penalizing. A small (possibly very small) value is necessary to avoid chain reactions in the self-contacting bodies but the same value will apply also to ordinary bodies, so that the extent of the refined zone will be dictated and no longer freely selectable for such bodies.

An improvement consists in allowing *two different* scaling factors:

- A scaling factor for ordinary (not self-contacting) bodies, ϕ_n, with a default value of 1.0, which can be changed (if necessary) by the optional `SCAL phin` keyword in the `PINB` directive.
- A scaling factor for self-contacting bodies, ϕ_s, with a default value of 0.55, which can be changed (if necessary) by the optional `SCAS phis` keyword in the `PINB` directive.

3 Numerical examples

We consider a series of numerical examples in order to test the models proposed in the previous Sections. All input files for these examples are listed in the Appendix.

3.1 Simple adaptivity-driven contact test

A simple 2D impact test between two elasto-plastic metal blocks is performed. The two blocks have opposite initial velocities of 100 m/s and collide when the gap existing at the beginning of the calculation gets closed. The calculations performed are summarized in Table 3.

Case	Mesh	Description
ADPI05	Q42L	"Manual" mesh refinement by `INIT ADAP SPLI` directive
ADPI04	Q42L	"Manual" mesh refinement by `PLAY ADAP SPLI` directive

Table 3: Calculations for the simple adaptivity-driven contact test.

The initial mesh is extremely coarse. It consists of just one quadrilateral element measuring 1×1 units for each block. Of course this not particularly realistic, but the test just aims at verifying the basic mesh adaptation mechanism in the presence of contact modelled by pinballs.

At the initial time, the mesh is adaptively refined by suitable commands up to a hierarchy level of 3 near the two faces of the blocks that will undergo contact. This generates a graded mesh with 8 Q4GL "smaller" elements along the contacting faces, each with a side of 0.125 units.

In the input file specification, the user provides only the information concerning the *base* mesh. To model contact between the two blocks, one (parent) pinball is embedded in each block. Each pinball is assigned to a different "body" (by the `PINB BODY` directive) so that contact between the two bodies is checked for:

```
LINK COUP
     PINB BODY LECT 1 TERM
          BODY LECT 2 TERM
```

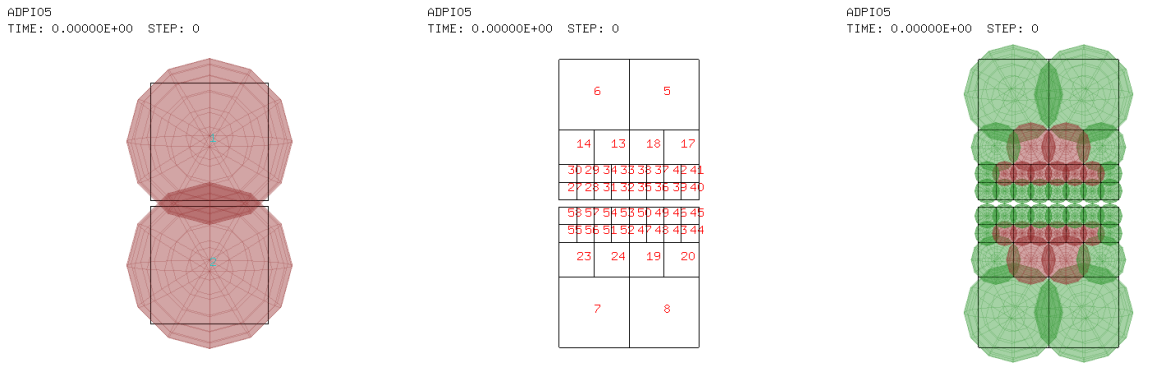
3.1.1 ADPI05

In this model, the mesh is refined during the initialization phase (by means of the `INIT ADAP SPLI` directive). This requires using the actual element indexes and would not be realistic in even slightly more complex applications, but it is acceptable for the purpose of this simple test:

```
INIT VITE 2 -100 LECT 1 PAS 1 4 TERM
     VITE 2  100 LECT 5 PAS 1 8 TERM
     ADAP SPLI LEVE 2 LECT 1 2 TERM
          SPLI LEVE 3 LECT 3 4 9 10 TERM
          SPLI LEVE 4 LECT 11 12 15 16 21 22 25 26 TERM
```

The resulting mesh at step 0, i.e. at the beginning of the transient calculation, is shown in Figure 8(a). The (parent) pinballs that *would be* embedded in the two base elements, if these would *not* be refined, are also shown (note that in that case contact would occur already at step 0).

Figure 8(b) shows the refined mesh at step 0 with the element indexes in red, and 8(c) shows the (parent) pinballs which are automatically inserted in the refined mesh (active descendents of the base mesh) by the strategy illustrated in the previous Section.



(a) Base mesh and pinballs (b) Refined mesh (c) Refined pinballs

Figure 8: Initial mesh refinement in case ADPI05.

The initial gap between the two blocks is computed so that no contact between the pinballs associated with the refined elements occurs at step 0, but it will occur at step 1 as the blocks have slightly moved towards each other. This is illustrated in Figure 9. By default, a parent pinball encompasses all nodes of the associated element. Therefore, the radius R of the pinball equals the semi-diagonal of the (regular square) element:

$$R = \frac{1}{2}\sqrt{2h^2} \tag{6}$$

where h indicates the side of the square. The pinball circle "protrudes" from the element shape by an "excentricity":

$$e = R - \frac{h}{2} \tag{7}$$

In the present case, contact between the two pinballs starts when the distance between the elements reaches the value:

$$d = 2e = 2R - h \tag{8}$$

For $h = 0.125$ we obtain $R = 8.83883 \times 10^{-2}$, $e = 2.58883 \times 10^{-2}$ and $d = 5.17767 \times 10^{-2}$. To avoid contact at the initial time, we choose a slightly larger initial gap: $g = 5.2 \times 10^{-2}$.

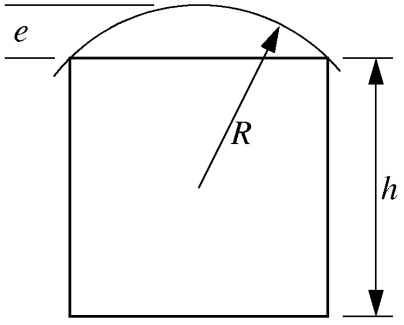
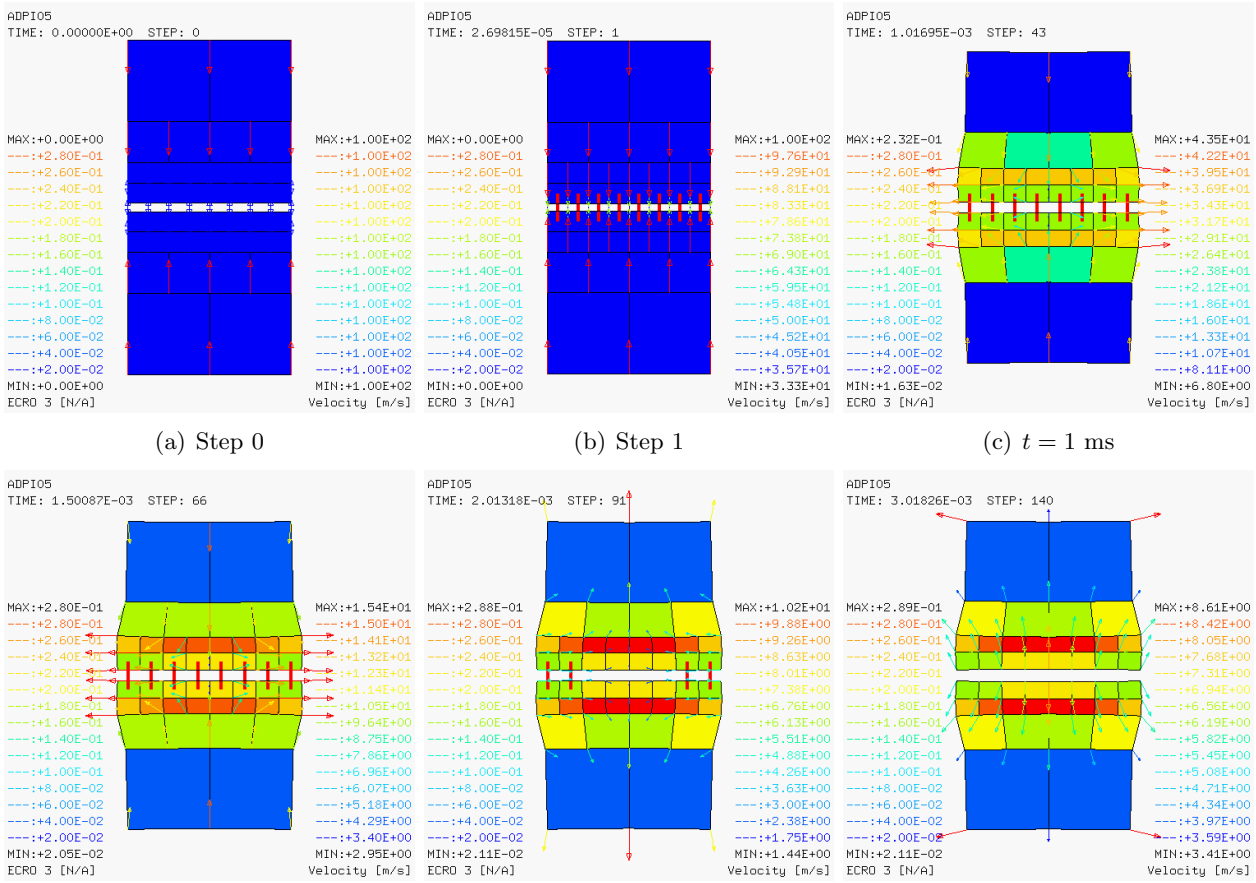


Figure 9: Computing the initial gap in case ADPI05.

The results calculation ADPI05 are illustrated in Figure 10. The colors indicate the level of plastification in the material, the arrows indicate the nodal velocities and the red thick lines (when present) indicate the "joints" between contacting pinballs (such lines join the centers of each couple of contacting pinballs).

In 10(a) one can see the situation at the initial time, when no contact occurs. First contact is detected at step 1, see 10(b), and then contact continues until about 1.5 ms inducing progressive plastification of the two blocks as shown in 10(c) and 10(d). At 2 ms (see Figure 10(e)) rebound starts from the center of the contacting zone and at the final time the two bodies are slowly rebounding, see 10(e). Since the material is elasto-plastic, much of the initial kinetic energy is absorbed in the material plastification process and only a fraction of this energy remains available as elastic waves which provoke the rebound of the two bodies.



(a) Step 0 (b) Step 1 (c) $t = 1$ ms

(d) $t = 1.5$ ms (e) $t = 2$ ms (f) $t = 3$ ms

Figure 10: Results of test case ADPI05.

3.1.2 ADPI04

This is a variant of case ADPI05 where a different approach for refining the initial mesh is taken. Instead of using the `INIT ADAP` directive, the mesh is refined by issuing "interactive" commands `ADAP SPLI` at the initial step. Interactive commands are meant to be executed from a console when the EPX code is launched in interactive execution mode (as specified by the `CONV WIN` directive included in the first part of the input file). However, by including such commands within a `PLAY` ... `ENDPLAY` pair of lines in the input, they are interpreted in "batch" (non-interactive) mode:

```
PLAY
 ...
ADAP
  SPLI 1
  SPLI 2
  SPLI 3
```

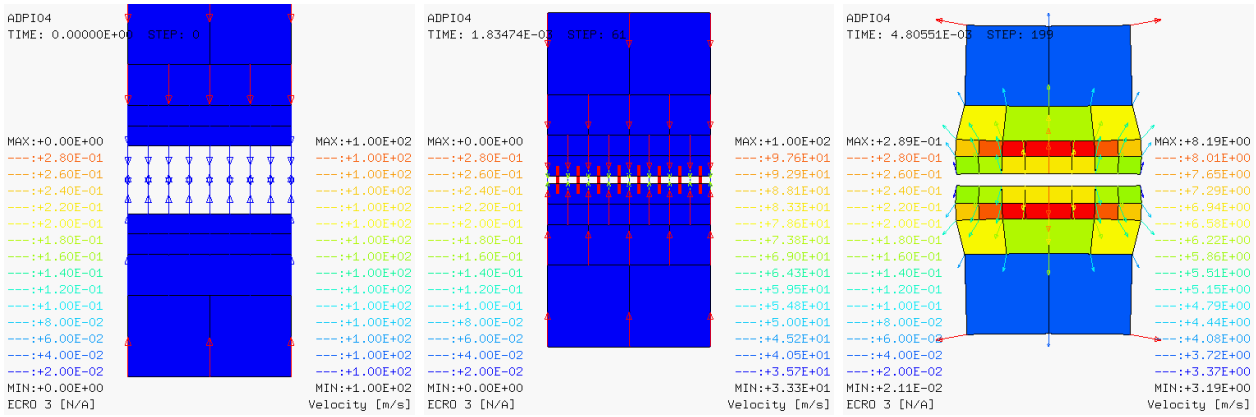
```
  SPLI 4
  SPLI 9
  SPLI 10
  SPLI 11
  SPLI 12
  SPLI 15
  SPLI 16
  SPLI 21
  SPLI 22
  SPLI 25
  SPLI 26
TERM
 ...
ENDPLAY
```

The other interactive commands present in the input file in the `PLAY` ... `ENDPLAY` sequence (and which are omitted in the above file snippet) are used to pilot the calculation and to produce an animation of the computed results (see the User's manual [1] for details).

Note that in this case the commands are *not* executed during the initialization phase (unlike in the previous test, where the `INIT` directive was used to refine the mesh), but *after* the 0 step (i.e. the initial equilibrium of the body) has been evaluated. This has consequences on the results. In fact, if one would use the same initial gap as in the previous test, contact would be detected between the two "big" parent pinballs at step 0 (see Figure 8(a)) *before* the mesh is refined, thus leading to completely different results from the previous simulation.

In order to obtain the same contact behaviour as in the previous test, a larger gap G must be initially assumed between the two bodies. The value to take for G is 8 times the one of the previous calculation, i.e. $G = 8g = 41.4214 \times 10^{-2}$. We assume here $G = 41.5 \times 10^{-2}$ in order to avoid contact at 0 step (between the unrefined pinballs), see Figure 11(a).

Immediately after step 0 the mesh is refined, so the large pinballs are desactivated, being replaced by the smaller ones shown in Figure8(c). Therefore, contact does not occur in this simulation until the gap has reduced to the value it had in the previous calculation at step 1. This requires after 61 steps and about 1.8 ms of "free flight" of the two bodies, as shown in Figure 11(b). The final situation at 4.8 ms is similar to the one in the previous solution at 3 ms, see Figure 11(c). This solution is very similar to case ADPI05, only shifted in time by 1.8 ms, due to the larger initial gap.



(a) Step 0 (b) Step 61, $t = 1.8$ ms (c) $t = 4.8$ ms

Figure 11: Results of test case ADPI04.

3.2 Blast-loaded plate test

We consider now a more realistic example of adaptivity in combination with contact. This is the simulation of a real experiment that is being conducted at NTNU Trondheim. A thin metallic square plate is clamped along its perimeter by a stiff bolted flange and is loaded in its central part by the blast wave resulting from the detonation of a high explosive charge located nearby. With certain values of the charge mass and distance, the plate is completely teared apart along the boundary of the

exposed area and flies away, bending upon itself and forming a characteristic "X"-shaped macroscopic fragment.

Contact occurs in various parts of the numerical model. First, between the plate and the frame and between the plate and the bolts. These contats are simulated by the `GLIS` model of EPX, which is optimally suitable for smooth contact between solids and includes friction modelling. Second, self-contact also accurs in the late phase of the experiment when the detached plate fragment bends up upon itself so that several parts of the plate fragment touch one another. For this type of contact, the pinball model is best used. At the moment, friction is not available in the pinball model but this is expected to play no role since the self-impact occurs normally to the involved surfaces.

In order to capture plate deformation and localized failure with the necessary accuracy and resolution, threshold-based adaptivity is activated in the central part of the plate, exposed to the blast wave, which will form the free flying fragment after rupture. Since pinballs are attached to this zone of the mesh in order to treat self-contact, this is a good test for the adaptivity-driven contact models presented above. The calculations performed for this test are summarized in Table 4.

Case	Mesh	Description
A0BG02	CUB8 and Q4GS	Threshold-driven adaptivity and self-contact by pinballs

Table 4: Calculations for the blast-loaded plate test.

3.2.1 A0BG02

Only the adaptivity- and (self) contact-related directive of the EPX input file are shortly commented below. The complete input file is listed in the Appendix, as usual.

```
A0BG02
ECHO
!CONV WIN
CAST mesh
EROS 1.0
```

The `EROS 1.0` directive activates element erosion when complete failure of an element occurs, i.e. when *all* of its Gauss points have failed. Of course, this concerns only elements associated with a material which admits failure.

```
ADAP THRS ECRO 11 TMIN 0.1 TMAX 0.5 MAXL 3
          LECT pinada TERM
```

The `ADAP THRS` directive introduces threshold-based adaptivity. Here the monitored quantity is ECR(11), which is the damage value for the `VPJC` material used to model the plate (see below). A maximum refinement level of 3 (thus two refinements) is prescribed as the monitored qiuantity (the damage in this case) passes between 0.1 (minimum threshold value `TMIN`) and 0.3 (maximum threshold value `TMAX`). Note that these values are much lower than 1.0 (full damage). The reason is that we want to *anticipate* local failure by refining the mesh well in advance, in order to capture the rupture in an accurate manner. Refining the mesh *after* rupture has started would not be very useful. The threshold model is applied to the central (exposed) part of the plate (`pinada`), which is also endowed by pinballs in order to treat self-contact in the macro fragment (see below).

```
MATE VPJC RO 2700.0 YOUN 70.0E9 NU 0.3 ELAS 80.0E6 mxit 500
          QR1 49.3E6 CR1 1457.1 QR2 5.2E6 CR2 121.5
          PDOT 5.E-4 C 0.014 TQ 0.9 CP 910.0
          TM 893.0 M 0.0 DC 1.0 WC 44.6E6
     LECT plate _q4gs TERM
```

The material model `VPJC`, that is a visco-plastic Johnson-Cook model, is applied to the plate. This material model has the notion of damage. This allows its use in conjunction with threshold-based adaptivity to track failure.

```
LINK COUP SPLT NONE SOLV PARD
     BLOQ 123 LECT bloc TERM
     BLOQ 1 LECT symx TERM
     BLOQ 2 LECT symy TERM
     BLOQ 56 LECT symxr TERM
     BLOQ 46 LECT symyr TERM
     PINB SELF MLEV 0 LECT pinada TERM
     GLIS 2 FROT MUST 0.1 MUDY 0.1 GAMM 0 ! Contact surface #1
            PGAP 0.4E-3
            MAIT LECT lframe TERM
            PESC LECT plateN  TERM
            FROT MUST 0.1 MUDY 0.1 GAMM 0 ! Contact surface #2
            PGAP 0.4E-3
            MAIT LECT uframeb TERM
            PESC LECT plateN   TERM
```

Boundary conditions are relatively complex in this test. Note the use of the pinballs (`PINB`) to define a self-contacting object (`SELF`) coincident with the exposed part of the plate (`pinada`). The `GLIS` contact model is used to treat the other contacts present, including the friction. This is a good exemple of the combination of two very different contact models in the same calculation.

The simulation until $t = 10$ ms takes 69 369 time steps and 4 040 s of CPU time on a laptop PC, by using only one processor (sequential version of EPX).

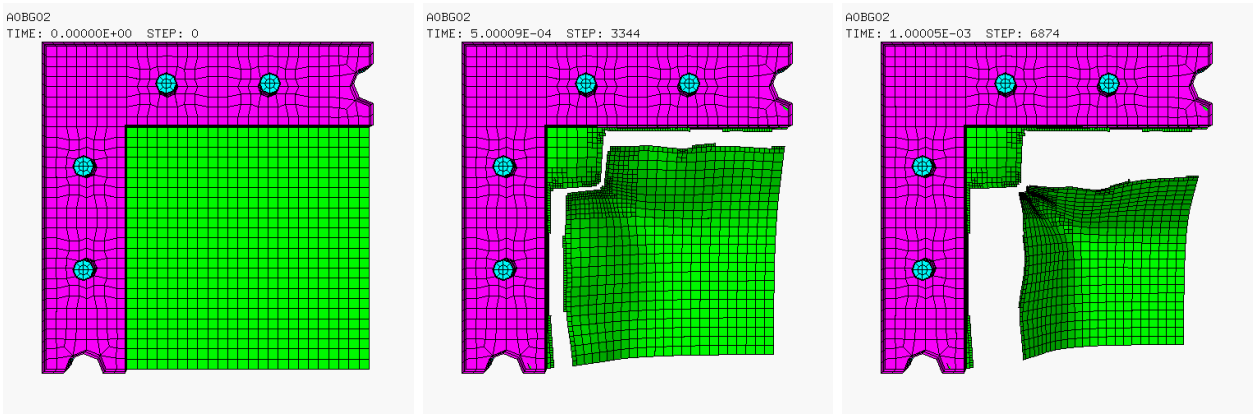
Some general results of this calculation are illustrated in Figure 12. Only 1/4 of the plate is represented in the numerical model, thanks to symmetry. In 12(a) the initial geometry ($t = 0$) is represented. The situation at $t = 1$ ms is depicted in 12(b), where the main fragment has just detached from the rest of the plate. In 12(c) one can see the final configuration at $t = 10$ ms. Several self-conatcts occur during the free-flying phase of the main fragment. Without self-contact the fragment parts would penetrate into one another thus leading to a non-physical shape of the residual fragment.

(a) $t = 0$ (b) $t = 1.0$ ms (c) $t = 10.0$ ms

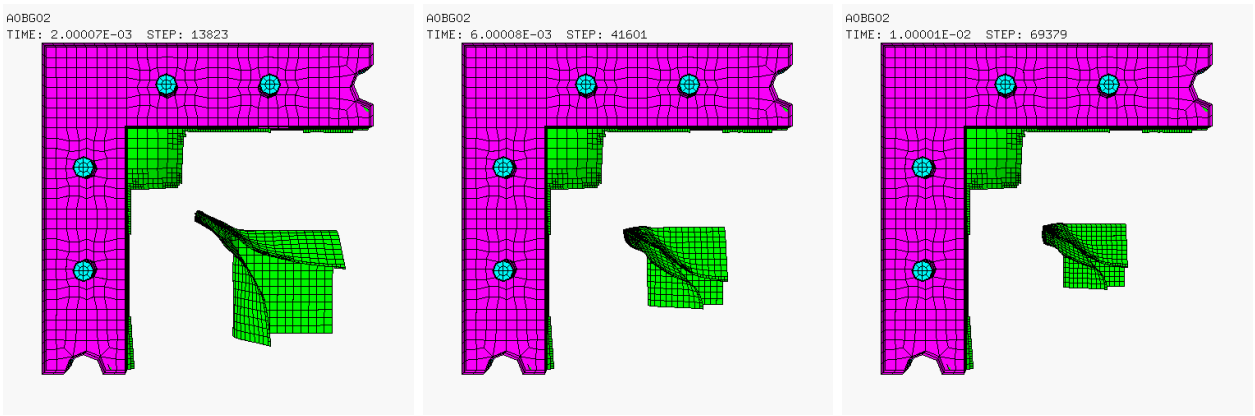
Figure 12: Results of test case A0BG02 (general view).

More detailed results of the same calculation are presented in Figure 13. Picture 13(a) shows the initial configuration ($t = 0$) from a viewpoint normal to the plate. In 13(b) one sees the detachment of the main fragment, at $t = 0.5$ ms. Note the localized refinement of the plate mesh near the rupture zone by means of threshold-driven adaptivity. The threshold is set on ECR(11) of the plate material (`VPJC`), which is a measure of the damage in the material. Although not shown in the picture, the pinballs embedded in the central part of the plate are also refined by constantly following the plate refinement. The other images 13(c) to 13(f) show the free flight of the fragment with the self-contact by pinballs preventing non-phisical interpenetrations. The fragment looks smaller and smaller in the perspective view, as its distance from the plate increases.

Figure 14 shows a symmetrized view (full model) of the plate simulation at various time instants, including the material damage represented by a scale of colors. Completely damaged parts of the plate have been eroded (`EROS 1.0`) and are not visible.

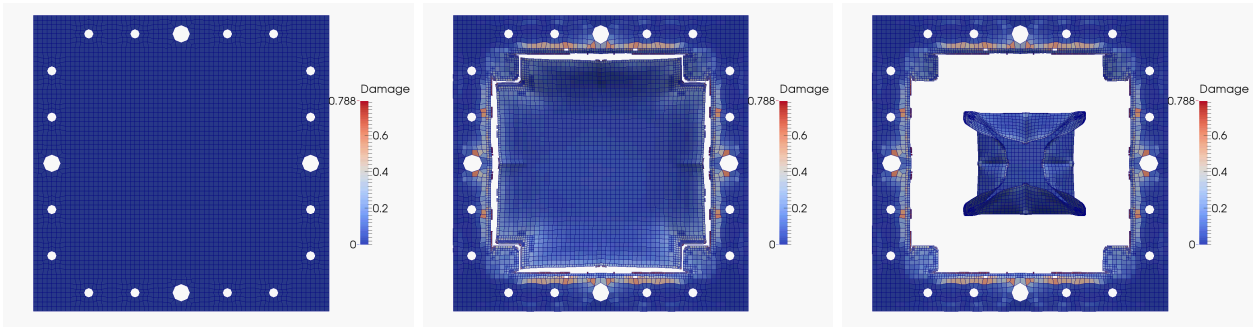


(a) $t = 0$ (b) $t = 0.5$ ms (c) $t = 1.0$ ms



(d) $t = 2.0$ ms (e) $t = 6.0$ ms (f) $t = 10.0$ ms

Figure 13: Results of test case A0BG02 (details).



(a) Initial time (b) Fragment detaches (c) Fragment's free flight

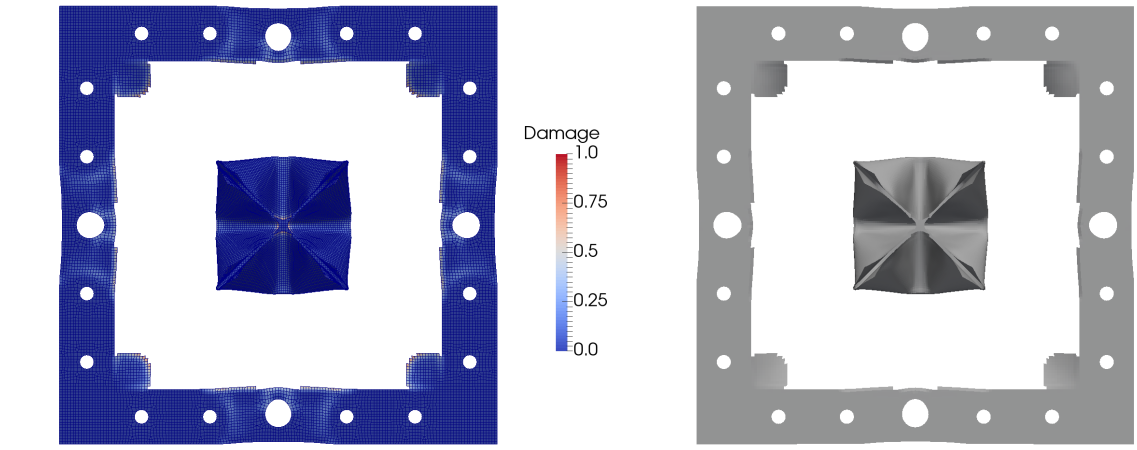
Figure 14: Symmetrized results of test case A0BG02 (full model view).

Finally, some comparisons between the computed and experimental results are made in Figure 15. In the top row 15(a) and 15(b) present the final shape of the ruptured plate and of the main fragment, with or without FE mesh lines. In the bottom row 15(c) depicts the two parts of the broken plate, while 15(d) and 15(e) show details of the real plate central fragment (front and rear views). The agreement between simulation and experiment is quite good.

3.3 Simple contact-driven adaptivity test

Let us now consider a very simple example of contact-driven adaptivity. We take a geometry similar to the one sown if Figure 11. The two bodies are modelled as linear elastic by means of a VM23 material with a very high elastic limit.

The base mesh is very coarse, consisting of only 6 quadrilaterals in the projectile (upper body) and



(a) Final with mesh

(b) Final without mesh

(c) Broken plate

(d) Fragment (front)

(e) Fragment (rear)

Figure 15: Comparison of test case A0BG02 with experimental results.

5 quadrilaterals in the target (lower body). These represent the parts of Figure 11 near the surfaces that will come into contact.

The triangular elements in the projectile of Figure 11 are not included in the model because at the moment quadrangles and triangles cannot be merged in the same adaptive calculation (the EPX adaptivity model will have to be completed in order to allow for this).

The calculations performed are summarized in Table 5.

Case	Mesh	Description
PIAD01	Q42L	Optional `NOUN` keyword prevents mesh un-refinement after contact
PIAD02	Q42L	Mesh un-refinement is allowed
PIAD03	Q42L	Hierarchic pinballs (`MLEV 2`), *not* adaptive
PIAD04	Q42L	Hierarchic pinballs (`MLEV 3`), adaptive

Table 5: Calculations for the simple contact-driven adaptivity test.

3.3.1 PIAD01

In this simulation, the mesh is automatically refined by anticipating possible contact, but mesh un-refinement is inhibited by the `NOUN` optional keyword. The complete contact directive reads:

```
LINK COUP SPLT NONE
     PINB BODY MLEV 0 LECT target TERM
          BODY MLEV 0 LECT projec TERM
     ADAP LMAX 3 SCAL 1.0 NOUN
```

Contact-driven mesh adaptivity is activated by the `ADAP` keyword in the `PINB` directive. A maximum refinement level of 3 is chosen (`LMAX 3`) at the default scaling factor (`SCAL 1.0`, which could as

well be omitted).

Note that the above value of $L_{\max}$ refers to the maximum refinement level *of the elements* (adaptivity) $L^{\text{adap}}_{\max}$, and not of the pinballs (contact) $L^{\text{pinb}}_{\max}$. This distinction is unfortunate and is only needed due to historical reasons: the pinball models was developed and implemented in EPX long before the adaptivity model. The relation between the levels is as follows: a base (not refined) element in adaptivity has by convention $L^{\text{adap}} = 1$, while a base (parent) pinball in the contact model has by convention $L^{\text{pinb}} = 0$. Thus, it should be kept in mind that:

$$L^{\text{adap}} = L^{\text{pinb}} + 1 \tag{9}$$

It seems preferable and more consistent with other adaptivity directives of EPX to use the `LMAX` keyword in the `PINB ... ADAP` directive to define $L^{\text{adap}}_{\max}$ rather than $L^{\text{pinb}}_{\max}$. In any case, it should be rarely necessary to use a hierarchic pinball method *in combination* with contact-driven adaptivity, so the level of the generated pinballs (attached to the smaller and smaller elements) will be zero, and the user can safely ignore this.

Some results of this test case are shown in Figure 16, where the mesh and the current (parent) pinballs are shown at various time instants. The color of each pinball indicates whether the pinball is *external* (i.e. it is located on the outer surface of a body), in green, or *internal*, in red.

Note that the base mesh is never visible in these images. In fact, the chosen parameters (initial gap between the bodies, size of the pinballs etc.) are such that the mesh is immediately refined (to level 2) due to contact anticipation at the initial time step ($t = 0$), see 16(a). It is highly desirable that the algorithm implementation allows for such a condition, because this is likely to be a common situation in practical applications.

In 16(b) the gap between the two bodies has narrowed and further refinement (to the level 3, which is the chosen maximum level) occurs locally, and then again in 16(c). Contact first occurs at about 10 ms, as shown in 16(d). Then, the target deform visibly and starts to oscillate elastically, as rebound starts, see 16(e) and 16(f).

As the two bodies are detached, no mesh unrefinement id performed as prescribed in the input file (`NOUN` keyword).

3.3.2 PIAD02

This calculation is a repetition of case PIAD01 but with mesh unrefinement enabled. The `NOUN` keywprd is removed from the input file.

Of course, the solution until first contact (i.e. during the mesh refinement phase) is identical to the one obtained in the previous case. These results are not shown for brevity. After contact the mesh is now progressively unrefined as the two objects come apart, as shown in Figure 17. The first unrefinement occurs at 28.0 ms, as shown in 17(a). Unrefinement continues during the subsequent steps until at 50.5 ms the mesh is completely unrefined and we can finally see the base mesh of the problem, see Figure 17(f).

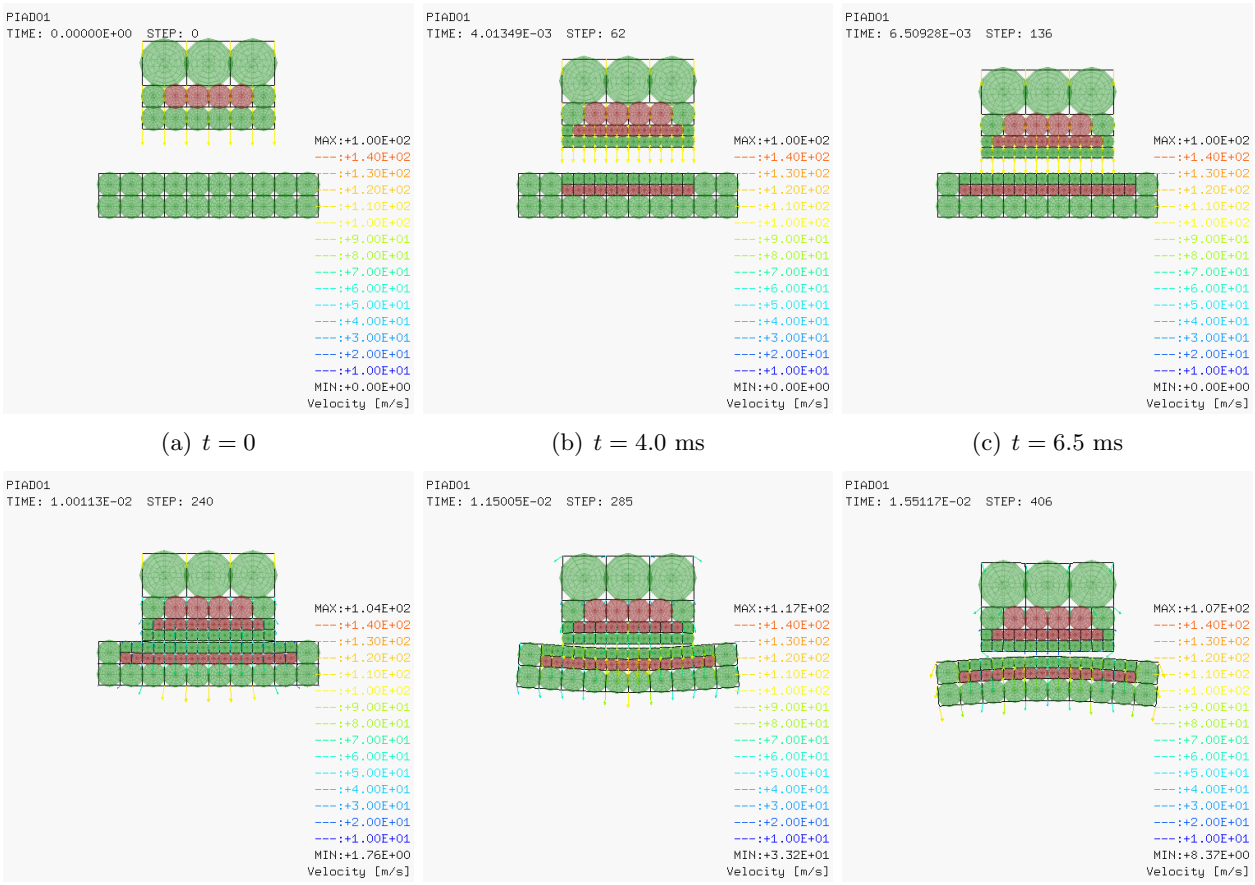
Figure 18 compares the (final) results of cases PIAD01 and PIAD02 at $t = 60$ ms. The solutions are similar. Some discrepancies are acceptable, given the great difference in mesh size between the two solutions in the rebound phase.

3.3.3 PIAD03

We want now to explore the use of *hierarchic* pinballs in combination with (or in alternative to) contact-driven adaptivity. First, we obtain a solution of the problem without adaptivity, but by prescribing hierarchich pinballs at `MLEV 2` (recall that this is the pinball level, not the element level).

The input directive for contact in this case becomes:

```
LINK COUP SPLT NONE
     PINB BODY MLEV 2 LECT target TERM
          BODY MLEV 2 LECT projec TERM
```



(a) $t = 0$ (b) $t = 4.0$ ms (c) $t = 6.5$ ms

(d) $t = 10.0$ ms (e) $t = 11.5$ ms (f) $t = 15.5$ ms

Figure 16: Results of case PIAD01.

Results of this calculation are presented in Figure 19. In 19(a) one can see the initial configuration. Since no adaptivity is prescribed, this corresponds to the *base* mesh of the previous simulations. The mesh is extremely coarse, definitely too coarse to get accurate results, but one should at least obtain a qualitatively correct response.

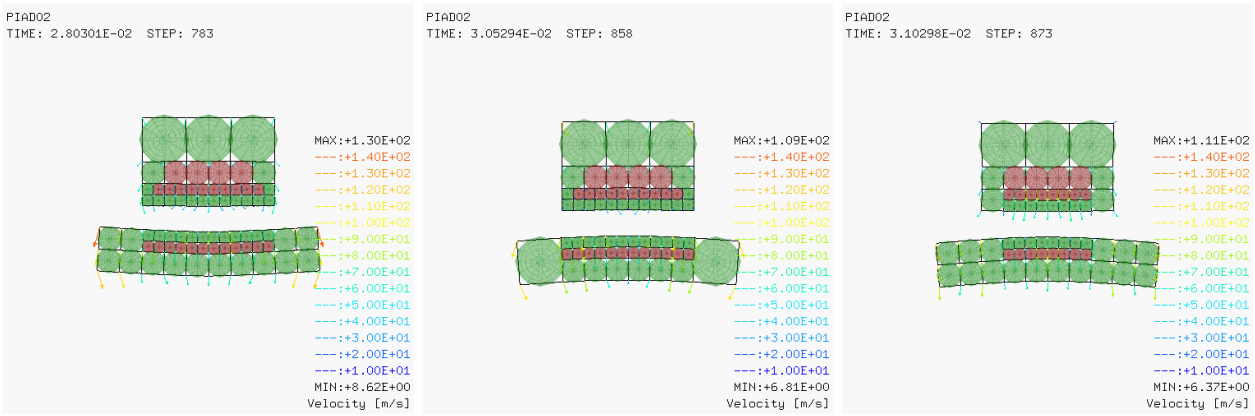
First contact occurs at $t = 9.0$ ms, see 19(b). The dark red spheres show the *contacting* hierarchic pinballs. These are the pinballs at the specified maximum level of the hierarchy ($L_{\max}^{\text{pinb}} = 2$) which happen to be in contact with one another at this time. The light green spheres represent the parent pinballs, as usual. Note that there are $3 \times 4 = 12$ contacting couples of pinballs (12 contacts), exactly like in cases PIAD01 (see Figure 16(d)) and PIAD02. The difference is that in those (adaptive) cases, the mesh had been refined so the contacting pinballs were 0-level (parent) pinballs, while in this case the contacting pinballs are sub-pinballs at a higer level. At $t = 10$ ms contact has already terminated and the rebound phase has started, see 19.

3.3.4 PIAD04

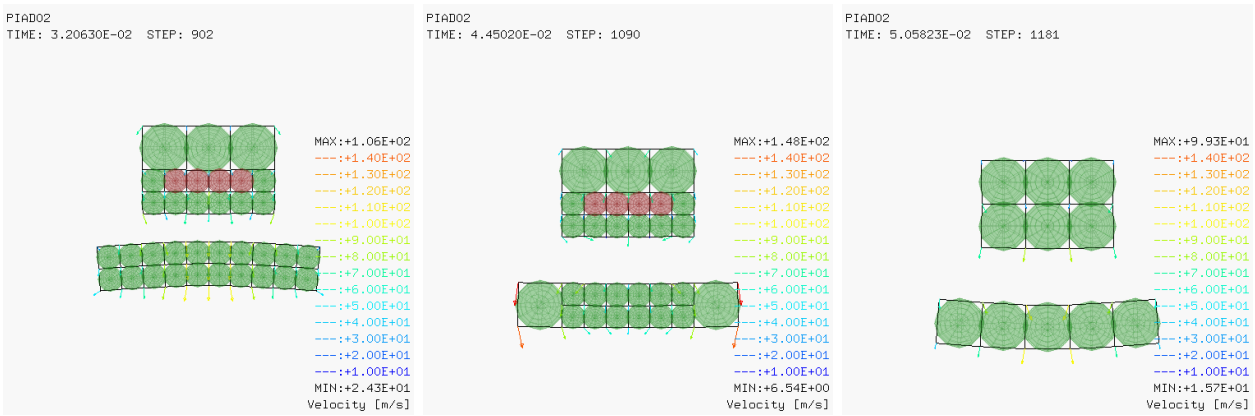
This is a repetition of case PIAD02 (adaptive) but by using hierarchic pinballs in combination with adaptivity. The maximum level of the hierarchic pinballs is set to 3 ($L_{\max}^{\text{pinb}} = 3$) instead of 0 while the maximum level for adaptivity is left to 3 ($L_{\max}^{\text{adap}} = 3$) like in the previous cases:

```
LINK COUP SPLT NONE
     PINB BODY MLEV 3 LECT target TERM
          BODY MLEV 3 LECT projec TERM
     ADAP LMAX 3 SCAL 1.0
```

Because of eq. (9), we expect to get once-refined pinballs within the twice-refined smallest elements along the contacting surfaces, thus twice the number of contacting pinballs with respect to the previous cases.

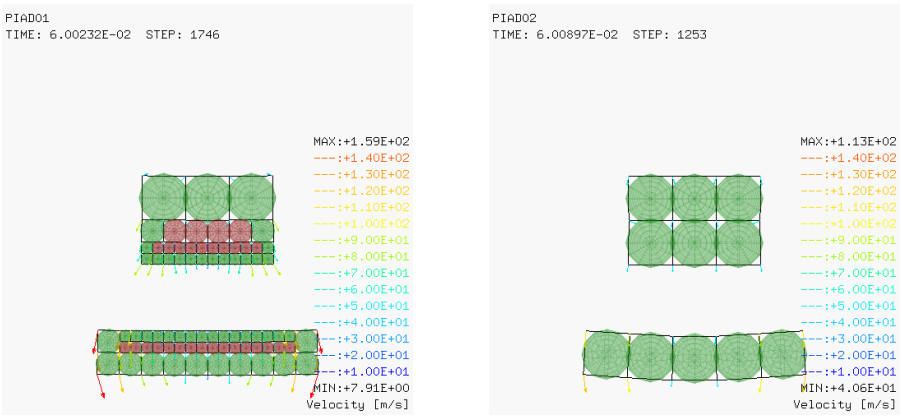


(a) $t = 28.0$ ms (b) $t = 30.5$ ms (c) $t = 31.0$ ms



(d) $t = 32.0$ ms (e) $t = 44.5$ ms (f) $t = 50.5$ ms

Figure 17: Results of case PIAD02 (rebound phase only).



(a) PIAD01, $t = 60$ ms (b) PIAD02, $t = 60$ ms

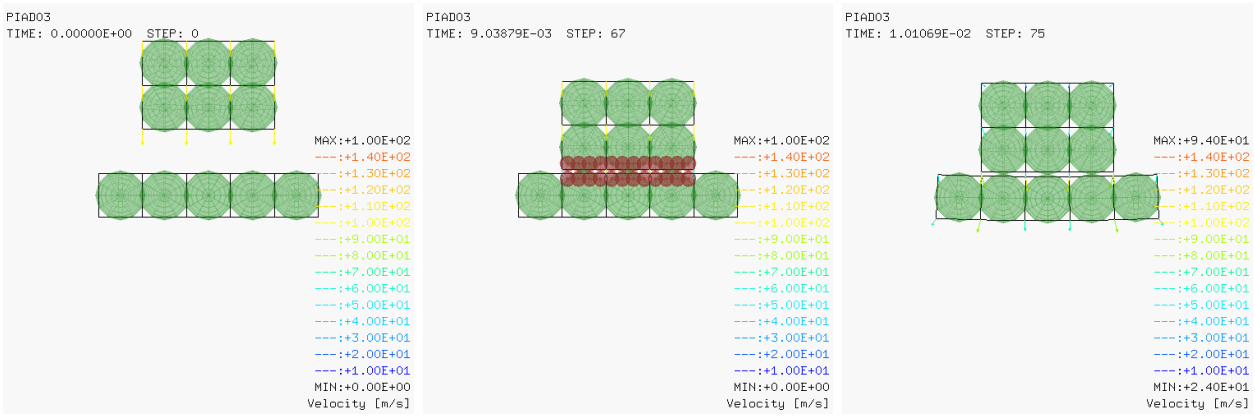
Figure 18: Results of cases PIAD01 and PIAD02 at the final time.

Results of this calculation are presented in Figure 20. As a matter of fact, when first contact is detected at 10.0 ms, there are 24 contacts, see the *dark* red spheres in 20(c) (the light red spheres indicate internal pinballs).

Figure 21 compares the (final) results of cases PIAD03 and PIAD04 at $t = 60$ ms.

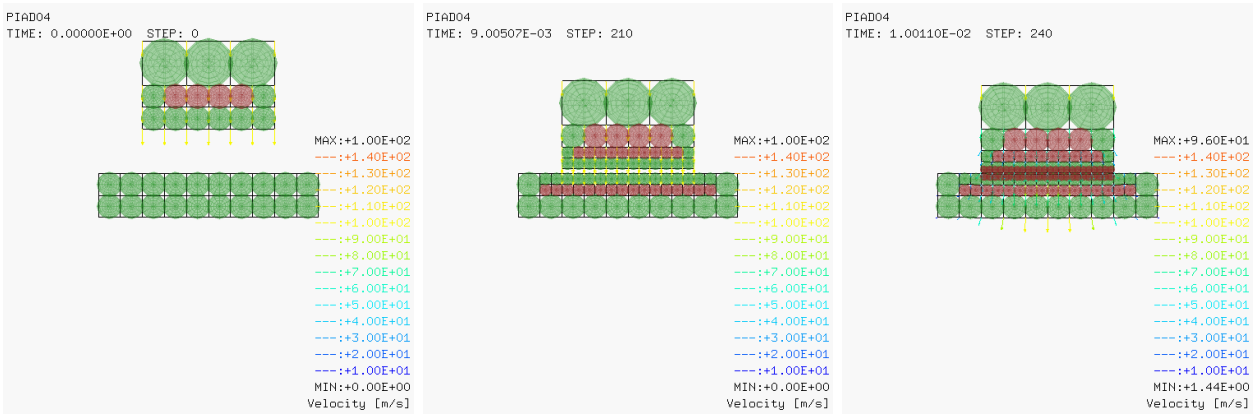
3.4 Combined threshold- and contact-driven adaptivity test

The next tests aim at verifying the combination of contact-driven adaptivity with another type of adaptivity. In fact, in some real applications the use of contact-driven adaptivity could be insufficient, if for some reason the mesh has to be adapted also *far* from the contacting surfaces.



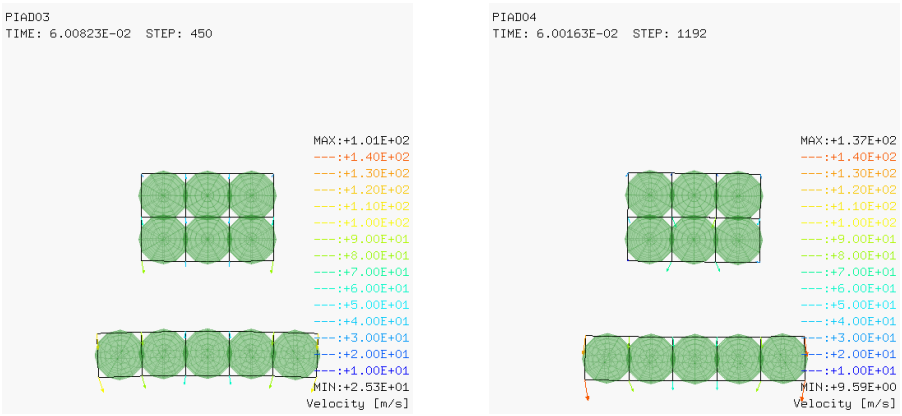
(a) $t = 0$ (b) $t = 9.0$ ms (c) $t = 10.0$ ms

Figure 19: Results of case PIAD03.



(a) $t = 0$ (b) $t = 9.0$ ms (c) $t = 10.0$ ms

Figure 20: Results of case PIAD04.



(a) PIAD03, $t = 60$ ms (b) PIAD04, $t = 60$ ms

Figure 21: Results of cases PIAD03 and PIAD04 at the final time.

As anticipated in a previous Section, some of the other adaptivity models present in EPX cannot be combined together at the moment of this writing. In particular, this is the case for the wave tracking directive (`WAVE`) and for the error indicators (`INDI`). However, other types of adaptivity could in principle be combined. In particular, the threshold-based adaptivity (`THRS`) is a good candidate since it is particularly simple. In fact, the threshold model in its current impementation is mainly used to track damage and failure of the structural domain (although other types of threshold could be prescribed in principle) and in the current version it refines but *never un-refines* the adapted mesh.

For this reason, it is less likely that it could enter in conflict with other more general adaptivity models present in the same calculation.

In principle, the new contact-driven adaptivity model has been programmed and implemented in such a way that it should be compatible with threshold-driven adaptivity. Of course, tests must be conducted to verify this useful property, which is absolutely essential in realistic applications.

The problem considered is the impact of a free-falling arch made of concrete onto a much stiffer metallic obstacle, blocked at its base. No reinforcement is present in the concrete, so the arch is expected to undergo severe failure and fragmentation. Contact-driven adaptivity is used for an accurate representation of the impact and (at the same time) threshold-based adaptivity anticipates and tracks material failure. Element erosion is also activated in order to allow separation of the macroscopic fragments expected from the failure of the concrete arch.

The numerical simulations performed are summarized in Table 6.

Case	Mesh	Description
PITH07	CUBE	Only threshold-based adaptivity
PITH08	CUBE	Only contact-based adaptivity
PITH08	CUBE	Combined threshold- and contact-based adaptivity

Table 6: Calculations for the free falling concrete arch impact test.

3.4.1 PITH07

The first simulation uses only threshold-based adaptivity (in the arch). No contact-driven adaptivity is activated. The input file for this test case is shortly commented below, as concerns the aspects relevant for the present report.

```
PITH07
ECHO
!CONV WIN
EROS 1.0
```

Erosion is activated (`EROS`) with the least possible fragile behaviour (1.0) of the material.

```
CAST mesh
LAGR TRID
DIME ADAP NPOI 2993 CUBE 2152 NPIN 2152 ENDA TERM
GEOM CUBE target projec TERM
COMP COUL VERT LECT target TERM
          TURQ LECT projec TERM
     NGRO 1 'bloc' LECT target TERM COND Y LT -2.99
ADAP THRS ECRO 13 TMIN 0.1 TMAX 0.5 MAXL 3
          LECT projec TERM
```

Threshold-based adaptivity (`ADAP THRS`) is set to refine the mesh in the projectile (the arch) up to level 3 (`MAXL 3`) as the monitored threshold quantity passes from 0.1 to 0.5. See comments on these parameters in a previous test case. The ECR(13) quantity for the `DPDC` material of the arch is the current damage, ranging from 0 (virgin material) to 1 (fully damaged material).

```
MATE LINE RO 8000. YOUN 1.D11 NU 0.3
          LECT target TERM
DPDC RO 2360 YOUN 40.1E+9 NU 0.2
     FC 75.E+6 DAGG 16.0E-3  VERS 8 EFVI
     EROD ENDT 0.99 ENDC 0.99 DVOL 0.3
          LECT projec _cube TERM
```

The metallic obstacle has a linear elastic material (`LINE`) while the `DPDC` model (Dynamic Plastic Damage Concrete) is used to represent the arch.

```
OPTI PINS EQVL
LINK COUP SPLT NONE
     BLOQ 123 LECT bloc TERM
     PINB BODY MLEV 0 LECT target TERM
          SELF MLEV 0 LECT projec TERM
```

```
INIT VITE 2 -100 LECT projec TERM
ECRI DEPL VITE ACCE FINT FEXT FLIA FDEC CONT ECRO TFRE 0.005
     FICH ALIC TFRE 0.0005
OPTI PAS AUTO NOTE LOG 1
     CSTA 0.5
     ADAP RCON
     PINS GRID DPIN 1.01
CALC TINI 0. TEND 0.02D0
FIN
```

The pinball model (`PINB`) is used to treat contact. The obstacle (`target`) is modelled as a body (`BODY`) since it may not fail and will undergo minimal deformations. The arch (`projec`), however, is modelled as a self-contacting body (`SELF`). This is to avoid inter-penetration of the arch fragments that are formed after failure of some parts of the arch and erosion of the corresponding elements.

This calculation runs until the chosen final time $t = 20$ ms in 2 316 time steps and 105 s of CPU time. Some results are presented in Figure 22. The overall behaviour seems fairly realistic. In the initial configuration 22(a) one can observe the base mesh. At $t = 1.5$ ms the threshold algorithm begins refining the mesh in zones where damage starts to build up, see 22(b). These should be zones where the concrete experiences some traction, because compression does not produce (or produces less) damage in this particular material. At $t = 2$ ms, see 22(c), the mesh has been further refined in these zones and the first cracks (localized element erosion) start to appear. Mesh refinement and cracking continue, see 22(d) and 22(e), until the calculation is stopped at $t = 20$ ms, see 22(f). Between 15 and 20 ms erosion has produced some macroscopic fragments, which enter in contact with one another (and with the obstacle) thanks to the self-contact directive of the pinball algorithm.

The final result, shown in 22(f), does *not* look fully realistic. In fact, no failure and no element refinement (thus indicating unrealistically low levels of damage) are observed in the central part of the arch, despite the fact that the upper-central part undergoes a clear change of curvature, which should lead to large local tractions in the central-upper part of the arch. However, this is more likely to be a problem in the material formulation (or possibly in the choice of DPDC material parameters, which are quite complex) than in the (threshold-driven) adaptivity algorithm. For this reason, investigation of the problem is outside the scope of the present work.

Since no contact-driven adaptivity is activated in this first simulation, the elements in the central part of the arch remain coarse (base mesh) and contact is detected by the associated (large-size) pinballs. The contact seems to work quite well, but the use of such large pinballs leaves a certain gap between the two bodies.

3.4.2 PITH08

This test case is similar to the previous one but only contact-driven adapativity (no threshold-driven adaptivity) is activated. In the input file, the `ADAP THRS` directive is completely removed and the `ADAP` sub-directive is added in the contact definition by pinballs:

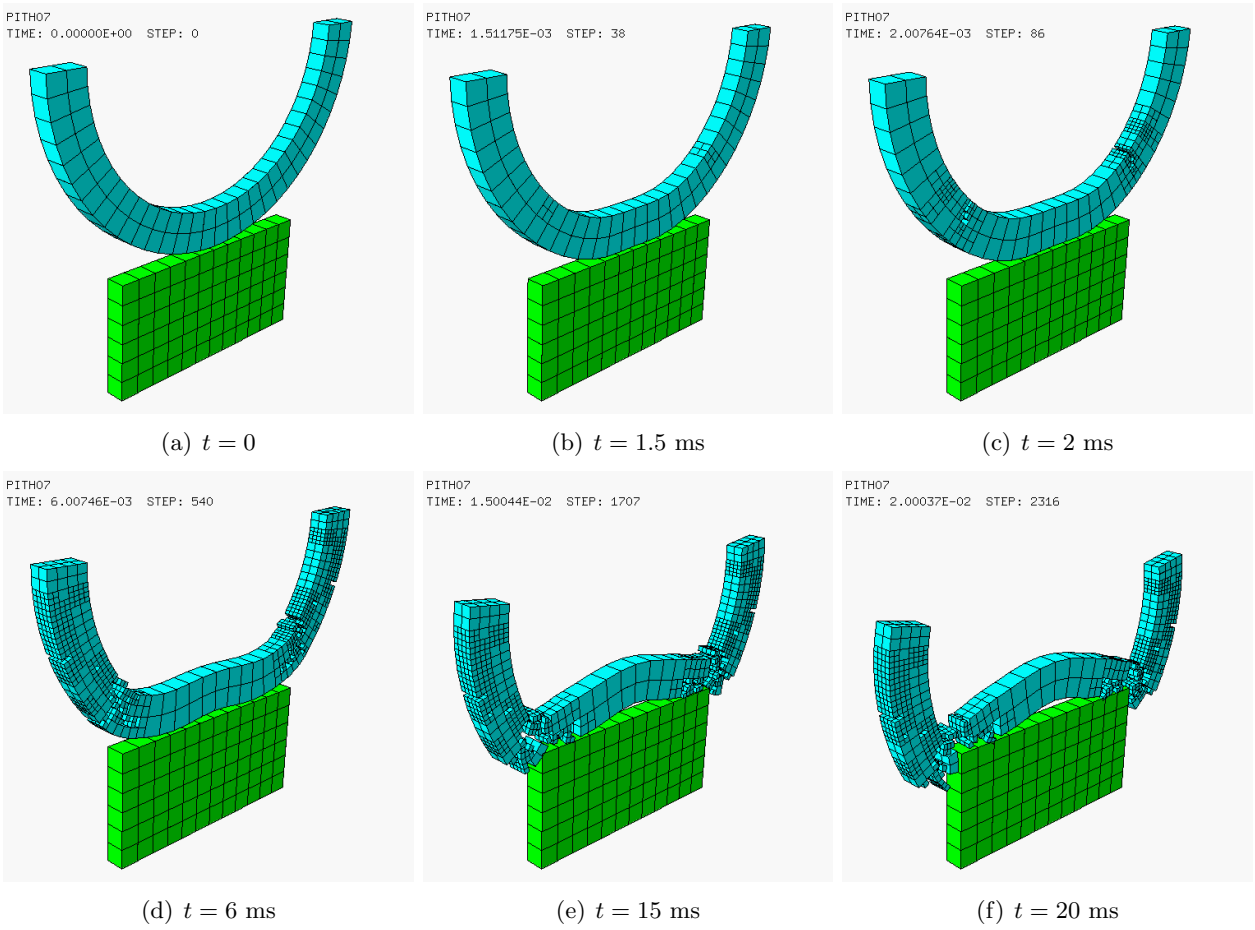
```
LINK COUP SPLT NONE
     BLOQ 123 LECT bloc TERM
     PINB BODY MLEV 0 LECT target TERM
          SELF MLEV 0 LECT projec TERM
          ADAP LMAX 3 SCAL 1.0 SCAS 0.55
```

This calculation fails at $t = 13.1$ ms, after 1 470 time steps and 63 s of CPU time. Reason of the failure is a problem in the geometric routine FIND_CORFACP which is used to compute the so-called *corner star* of elements (in this case, hexahedra) attached to a given corner. The problem is under investigation. It might be due to combination of adaptivity with element erosion.

Some results of this calculation are presented in Figure 23 at the same times as in the previous case until $t = 6$ ms, and at the last stored time station which is at $t = 13$ ms.

3.4.3 PITH09

This is the simulation with combined threshold- and contact-driven adaptivity. Both the `ADAP THRS` (like in case PITH07) and the `ADAP` sub-directive of pinballs definition are present:



(a) $t = 0$ (b) $t = 1.5$ ms (c) $t = 2$ ms

(d) $t = 6$ ms (e) $t = 15$ ms (f) $t = 20$ ms

Figure 22: Results of case PITH07.

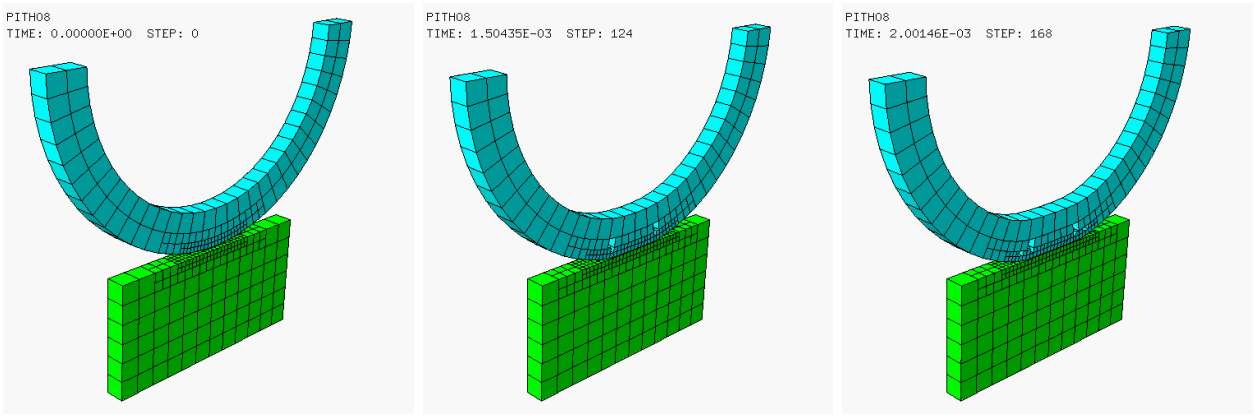
```
ADAP THRS ECRO 13 TMIN 0.1 TMAX 0.5 MAXL 3
          LECT projec TERM
MATE LINE RO 8000. YOUN 1.D11 NU 0.3
          LECT target TERM
DPDC RO 2360 YOUN 40.1E+9 NU 0.2
     FC 75.E+6 DAGG 16.0E-3  VERS 8 EFVI
     EROD ENDT 0.99 ENDC 0.99 DVOL 0.3
          LECT projec _cube TERM
OPTI PINS EQVL
LINK COUP SPLT NONE
     BLOQ 123 LECT bloc TERM
     PINB BODY MLEV 0 LECT target TERM
          SELF MLEV 0 LECT projec TERM
          ADAP LMAX 3 SCAL 1.0 SCAS 0.55
```

The solution of this test case becomes extremely slow starting at around 9 ms. Some of the concrete elements undergo very large distortions, but *without* failing and being eroded. Their stability step becomes very small and this slows down the entire calculation. This behaviour might be due to the use of an under-integrated element (CAR1), or to a problem in the DPDC material (either due to the chosen input parameters or in the material routine itself). It is strange, however, that this type of difficulties did not emerge in the previous two simulations.

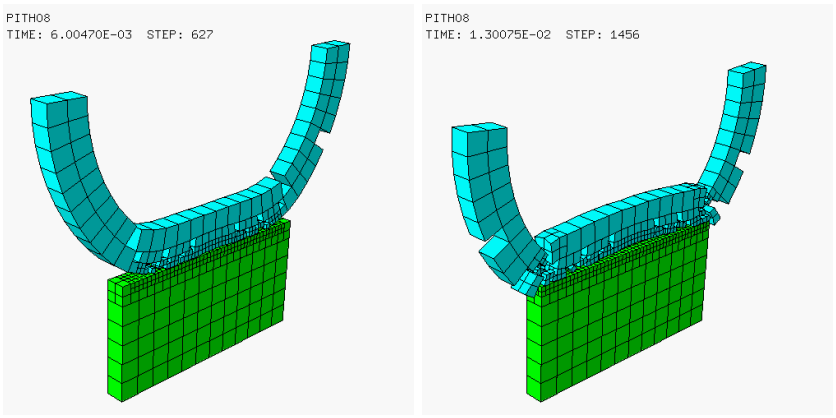
Since both such aspects are out of scope in the present work, an optional keyword `TFAI 3.E-6` is tentatively added in the `CALC` directive. This causes immediate failure of any *erodible* elements (i.e. elements associated with an erodible material, such as DPDC) as soon as their stability step falls below the chosen limit of 3 μs.

```
CALC TINI 0. TEND 0.02D0 TFAI 3.E-6
```

With this modification the calculation proceeds more quickly, but it fails at $t = 12.9$ ms, after 3 332 time steps and 710 s of CPU time. The reason for failure is a message from the DPDC material



(a) $t = 0$ (b) $t = 1.5$ ms (c) $t = 2$ ms



(d) $t = 6$ ms (e) $t = 13$ ms

Figure 23: Results of case PITH08.

routine, saying that the number of sub-steps is too large. Again, this might indicate that the concerned element should have perhaps been eroded before, but investigation of such aspects is out of scope here.

Some results of this calculation are presented in Figure 24 at the same times as in the previous case until $t = 6$ ms, and at the last stored time station which is at $t = 12.5$ ms.

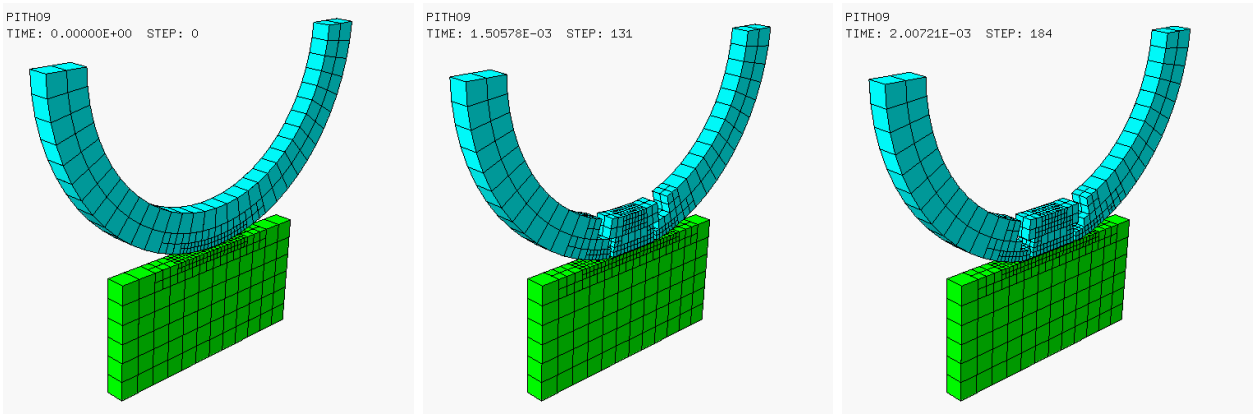
Another strange result of this calculation is that just before failure of the run the target deforms heavily near the contact zone, see Figure 24(e), a phenomenon that was not observed in the previous simulations. This might indicate an incipient instability of the calculation. The energy balance and even the mass balance become very bad (unlike in the previous solutions) and the contact forces are huge. For all this reasons, results of this calculation beyond, say, 10 ms or so are not reliable.

Figure 25 compares the three solutions PITH07, PITH08 and PITH09 at $t = 12.5$ ms. The solution with only threshold-driven adaptivity 25(a) runs smoothly but suffers from poor resolution in the contact zone, since the mesh is never refined there. The solution with only contact-driven adaptivity presents the wanted refinement near the contacting surfaces but the rest of the arch is never refined and so the falure of the arch is treated a bit too coarsely, see 25(b). The combined solution of 25(c) shows the potential for obtaining an optimal combination of both adaptation strategies, but the reason for numerical difficulties must be investigated and solved first.

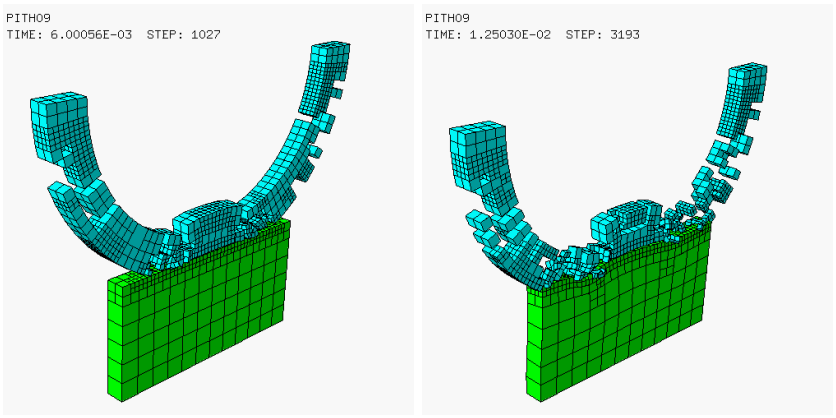
References

[1] EUROPLEXUS User's Manual, on-line version: http://europlexus.jrc.ec.europa.eu.

[2] Cast3m Software: http://www-cast3m.cea.fr/.

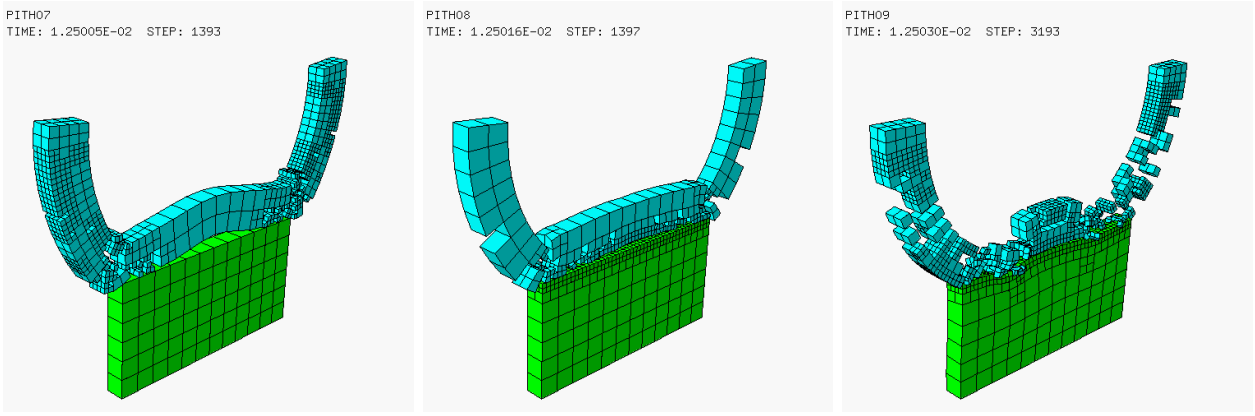


(a) $t = 0$ (b) $t = 1.5$ ms (c) $t = 2$ ms



(d) $t = 6$ ms (e) $t = 12.5$ ms

Figure 24: Results of case PITH09.



(a) Threshold (b) Contact (c) Threshold and contact

Figure 25: Comparison of solutions PITH07, PITH08 and PITH09 at $t = 12.5$ ms.

[3] J.O. Hallquist, G.L. Goudreau, D.J. Benson. *Sliding interfaces with contact-impact in large-scale Lagrangian computation.* Computer Methods in Applied Mechanics and Engineering **51** : 107–137, 1985.

[4] D.J. Benson, J.O. Hallquist. *A single surface contact algorithm for the postbuckling analysis of shell structures.* Report to the University of California at San Diego, CA, 1987.

[5] T. Belytschko, S.E. Law. *An Assembled Surface Normal Algorithm for Interior Node Removal in Three-Dimensional Finite Element Meshes.* Engineering with Computers **1** : 55–60, 1985.

[6] T. Belytschko, J.I. Lin. *A Three-Dimensional Impact-Penetration Algorithm with Erosion.* Computers and Structures **25** : 95–104, 1987.

[7] T. Belytschko, M.O. Neal. *Contact-Impact by the Pinball Algorithm with Penalty, Projection and Lagrangian Methods.* Computational Techniques for Contact, Impact, Penetration and Perforation of Solids, Winter Annual Meeting of the American Society of Mechanical Engineers, San Francisco, California **AMD 103** : 97–140, December 10–15, 1989.

[8] T. Belytschko, M.O. Neal. *The Vectorized Pinball Contact Impact Routine.* SMiRT-10 Conference, 1989.

[9] R.E. Sarwas. *Hidden Line Elimination and a Modified Pinball Algorithm for Finite Element Contact-Impact Problems with Shell Elements.* Master's Thesis, Northwestern University, Evanston, Illinois, December 1989.

[10] T. Belytschko, M.O. Neal. *Contact-Impact by the Pinball Algorithm with Penalty and Lagrangian Methods.* International Journal for Numerical Methods in Engineering **31** : 547–572, 1991.

[11] T. Belytschko, I.S. Yeh. *The Splitting Pinball Method for Contact-Impact Problems.* Computer Methods in Applied Mechanics and Engineering **105** : 375–393, 1993.

[12] E. Plaskacz, T. Belytschko, H.Y. Chiang. *Contact-Impact Simulations on Massively Parallel SIMD Supercomputers.* Argonne National Laboratory Report ANL/CP-76686, 1993.

[13] M. Lepareux, B. Schwab, A. Hoffmann, P. Jamet, H. Bung. *Un programme général pour l'analyse dynamique rapide – Cas des tuyauteries.* Colloque: Tendances Actuelles en Calcul des Structures, Bastia, 6–8 November, 1985.

[14] F. Casadei. *A Hierarchic Pinball Method for Contact-Impact in Fast Transient Dynamics.* VI Congresso Nazionale della Società Italiana di Matematica Applicata e Industriale (SIMAI 2002), Chia (Cagliari), Italy, 27–31 May 2002.

[15] F. Casadei. *A General Impact-Contact Algorithm Based on Hierarchic Pinballs for the EUROPLEXUS Software System.* Joint Research Center Report N. I.03.176, December 2003.

[16] F. Casadei. *Implementation of Fast Search Algorithms in EUROPLEXUS.* Technical Note, PUBSY N. JRC38086, 2007.

[17] F. Casadei. *Validation of the EUROPLEXUS Pinball Impact-Contact Model on an Indentation Problem.* Technical Note N. I.05.51, July 2005.

[18] F. Casadei, M. Larcher, G. Valsamos, B. Langrand. *Pinball-based Contact-Impact Model with Parabolic Elements in EUROPLEXUS.* Technical Note JRC Pubsy N. JRC89913, EUR Report 26629 EN, 2014.

[19] F. Casadei, M. Larcher, G. Valsamos, V. Faucher. *Implementation of Assembled Surface Normals and of a Penalty Contact Formulation in the Pinball Model of EUROPLEXUS.* Technical Note, PUBSY No. JRC90939, EUR 26714 EN, 2014.

[20] F. Casadei, M. Larcher, G. Valsamos. *A contact model based on generalized pinballs in EUROPLEXUS.* Technical Note in preparation, 2016.

[21] F. Casadei, P. Díez, F. Verdugo. *A Data Structure for Adaptivity in EUROPLEXUS.* JRC Technical Note PUBSY N. JRC60795, September 2010.

[22] F. Casadei, P. Díez, F. Verdugo. *Adaptivity in FE Models for Fluids in EUROPLEXUS.* JRC Technical Note PUBSY N. JRC61622, November 2010.

[23] F. Casadei, P. Díez, F. Verdugo. *Adaptive 3D Refinement and Un-refinement of 8-node Solid and Fluid Hexahedra in EUROPLEXUS.* JRC Technical Note PUBSY N. JRC63833, March 2011.

[24] F. Casadei, P. Díez, F. Verdugo. *Implementation of a 2D Adaptivity Indicator for Fast Transient Dynamics in EUROPLEXUS.* JRC Technical Note PUBSY N. JRC64506, April 2011.

[25] F. Casadei, P. Díez, F. Verdugo. *Further Development of 2D Adaptivity Error Indicators in EUROPLEXUS.* JRC Technical Note, PUBSY No. JRC66337, September 2011.

[26] F. Verdugo, P. Díez, F. Casadei. *Natural quantities of interest in linear elastodynamics for goal oriented error estimation and adaptivity.* Proceedings of the V International Conference on Adaptive Modeling and Simulation (ADMOS 2011), D. Aubry and P. Díez (Eds), Paris, France, 6–8 June 2011.

[27] F. Verdugo, P. Díez, F. Casadei. *General form of the natural quantities of interest for goal oriented error assessment and adaptivity in linear elastodynamics.* Submitted for publication in the International Journal for Numerical Methods in Engineering, DOI: 10.1002/nme, PUBSY No. JRC65788, July 2011.

[28] F. Casadei, G. Valsamos, P. Díez, F. Verdugo. *Implementation of Adaptivity in 2D Cell Centred Finite Volumes in EUROPLEXUS.* Technical Note, PUBSY No. JRC67859, December 2011.

[29] F. Casadei, P. Díez, F. Verdugo. *Implementation of Adaptivity in 3D Cell Centred Finite Volumes in EUROPLEXUS.* Technical Note, PUBSY No. JRC68168, December 2011.

[30] F. Casadei, P. Díez, F. Verdugo. *Testing Adaptivity in 2D Cell Centred Finite Volumes with the CDEM Combustion Model in EUROPLEXUS.* Technical Note, PUBSY No. JRC68333, December 2011.

[31] F. Casadei, P. Díez, F. Verdugo. *An algorithm for mesh refinement and un-refinement in fast transient dynamics.* International Journal of Computational Methods, DOI 10.1142/S0219876213500187, Vol. **10**, No. 4, pp. 1350018-1 / 1350018-31, 2013.

[32] F. Casadei, G. Valsamos, M. Larcher, A. Beccantini. *Combination of Mesh Adaptivity with Fluid-Structure Interaction in Europlexus.* Technical Note, PUBSY No. JRC89728, EUR Report 26617 EN, 2014.

[33] F. Casadei, M. Larcher, G. Valsamos. *Adaptivity with Simplex Elements in Europlexus.* Technical Note, PUBSY No. JRC89894, EUR Report 26630 EN, 2014.

[34] F. Casadei, M. Larcher, G. Valsamos. *Adaptivity in CEA's Fluid Elements in Europlexus.* Technical Note, PUBSY No. JRC89953, EUR Report 26632 EN, 2014.

[35] M. Larcher, F. Casadei, G. Valsamos. *Adaptivity in Shell/Beam/Bar Elements in Europlexus.* Technical Note, PUBSY No. JRC90456, EUR Report 26697 EN, 2014.

[36] V. Faucher, F. Casadei. *Adaptive mesh refinement with combined criteria for fast transient fluid-structure dynamics.* 5th International Conference on Computational Methods in Structural Dynamics and Earthquake Engineering, COMPDYN 2015, Crete, Greece, 25–27 May 2015.

[37] F. Casadei, V. Faucher, M. Larcher, G. Valsamos. *Combined Fluid and Structure Mesh Adaptivity with Fluid-Structure Interaction in EUROPLEXUS.* Technical Note, PUBSY No. JRC96043, EUR Report 27276 EN, 2015.

[38] F. Casadei, G. Valsamos, M. Larcher. *On the interpretation and post-processing of mesh-adaptive numerical results in EUROPLEXUS.* Technical Note, PUBSY No. JRC96386, EUR Report 27316 EN, 2015.

[39] F. Casadei, V. Faucher, M. Larcher, G. Valsamos. *Decoupled formulation of constraints on adaptive hanging nodes in EUROPLEXUS.* Technical Note, PUBSY No. JRC96807, EUR Report 27347 EN, 2015.

[40] F. Casadei. *Implementation of Fast Search Algorithms in EUROPLEXUS.* Technical Note, PUBSY No. JRC38086, October 2007.

Appendix — Input files

All the input files used in the previous Sections are listed below.

a0bg02.dgibi

```
'DEBPROC' pxbox3d x0*'FLOTTANT' y0*'FLOTTANT' z0*'FLOTTANT'
                  lx*'FLOTTANT' ly*'FLOTTANT' lz*'FLOTTANT'
                  dd*'FLOTTANT';
*
*-------------------------------------------------
* Generates a parallelepiped mesh with origin in point
* (x0,y0,z0), sides of length (lx,ly,lz) and density (mesh size) dd.
* The mesh consists of CUB8 hexahedral elements and is oriented
* along the global axes.
*
* Input :
* -----
*     x0,y0,z0 : coordinates of 'origin' of the box
*     lx,ly,lz : length of the box sides
*     dd : "density" (size) of the mesh (the same in all directions)
* Output :
* ------
*     box : mesh consisting of CUB8 hexahedra
*-------------------------------------------------
*
dens dd;
p1 = x0 y0 z0;
p2 = (x0 + lx) y0 z0;
p3 = (x0 + lx) (y0 + ly) z0;
p4 = x0 (y0 + ly) z0;
*
c1 = p1 d p2;
c2 = p2 d p3;
c3 = p3 d p4;
c4 = p4 d p1;
base = dall c1 c2 c3 c4 plan;
*
box = base volu tran (0 0 lz);
*
finproc box;
*-----------------------------------------------------------------------
'DEBPROC' pxrec3d x0*'FLOTTANT' y0*'FLOTTANT' z0*'FLOTTANT'
                  lx*'FLOTTANT' ly*'FLOTTANT'
                  dd*'FLOTTANT';
*
*-------------------------------------------------
* Generates a rectangle mesh with origin in point
* (x0,y0,z0), sides of length (lx,ly) and density (mesh size) dd.
* The mesh consists of QUA4 quadrilateral elements and is oriented
* along the global axes.
*
* Input :
* -----
*     x0,y0,z0 : coordinates of 'origin' of the box
*     lx,ly    : length of the box sides
*     dd : "density" (size) of the mesh (the same in all directions)
* Output :
* ------
*     box : mesh consisting of CUB8 hexahedra
*-------------------------------------------------
*
dens dd;
p1 = x0 y0 z0;
p2 = (x0 + lx) y0 z0;
p3 = (x0 + lx) (y0 + ly) z0;
p4 = x0 (y0 + ly) z0;
*
c1 = p1 d p2;
c2 = p2 d p3;
c3 = p3 d p4;
c4 = p4 d p1;
rect = dall c1 c2 c3 c4 plan;
*
finproc rect;
*-----------------------------------------------------------------------
*$$$$ PX4CIR3D
*
* Pour generer le maillage 3D (plan) d'un quart de cercle
* avec seulement des quadrilateres a 4 noeuds.
* Le quart de cercle est defini par les deux extremes
* d'un arc (de 90 degrees), par le centre du cercle
* et par un autre point qui definit l'axe de rotation
* (axe perpendiculaire au plan du cercle, passant pour son centre).
*
* Input:
* =====
* P1 = premiere extremite de l'arc
* P2 = deuxieme extremite de l'arc
* PC = centre de l'arc
* PZ = autre point de l'axe
* N  = nombre de mailles a generer sur chaque cote (doit etre pair)
* TOL= tolerance pour l'elimination des noeuds doubles
*
* Output:
* ======
* SUR = objet MAILLAGE d'elements de type QUA4
* IER = 0: pas d'erreur, .NE.0: erreur dans la generation de SUR
*
'DEBPROC' PX4CIR3D P1*'POINT' P2*'POINT' PC*'POINT' PZ*'POINT'
                  N*'ENTIER' TOL*'FLOTTANT';
*-------------------------------------------------------------
*
ier=0;
n2 = n / 2;
p0 = 0 0 0;
pm1 = p1 plus p0;
depl pm1 tour 45 pc pz;
pm2 = 0.5*(pc plus p2);
pm3 = 0.5*(pc plus p1);
pm = 0.5*(pc plus pm1);
c1a = cerc n2 p1 pc pm1;
c1b = cerc n2 pm1 pc p2;
c2a = droi n2 p2 pm2;
c2b = droi n2 pm2 pc;
c3a = droi n2 pc pm3;
c3b = droi n2 pm3 p1;
c4a = droi n2 pm pm1;
c4b = droi n2 pm pm2;
c4c = droi n2 pm pm3;
sur1 = dall plan c4c c3b c1a (inve c4a);
sur2 = dall plan c4a c1b c2a (inve c4b);
sur3 = dall plan c2b c3a (inve c4c) c4b;
sur = sur1 et sur2 et sur3;
*
elim tol sur;
*
'FINPROC' sur ier;
*-----------------------------------------------------------------------
*
opti echo 1;
opti dime 3 elem cub8;
opti trac psc ftra 'a0bg02_mesh.ps';
opti sauv form 'a0bg02.msh';
*
*dd = 0.0025;
dd = 0.00625;
dens dd;
tol = 0.1*dd;
*
* Plate
*
pla1 = pxrec3d 0.0    0.0    0.0 0.15   0.15   dd;
pla2 = pxrec3d 0.15   0.15   0.0 0.05   0.05   dd;
pla3 = pxrec3d 0.0875 0.15   0.0 0.0125 0.05   dd;
pla4 = pxrec3d 0.03   0.15   0.0 0.0075 0.05   dd;
pla5 = pxrec3d 0.15   0.0875 0.0 0.05   0.0125 dd;
pla6 = pxrec3d 0.15   0.03   0.0 0.05   0.0075 dd;
trac qual cach (pla3 ET pla4);
*
p0 = 0 0 0;
***/*/*/*/*/cont = p0 d (0.025 0 0) d (0.025 0.0185 0) c 4 (0.025 0.025 0)
cont = p0 d (0.025 0 0) d (0.025 0.0185 0) c 2 (0.025 0.025 0)
       (0.0185 0.025 0) d (0 0.025 0) d p0;
trac qual cach cont;
qua1 = cont surf plan;
trac qual cach qua1;
qua2 = qua1 syme plan (0.025 0 0) (0.025 0.025 0) (0.025 0 0.025);
qua12 = qua1 et qua2;
qua34 = qua12 syme plan (0 0.025 0) (0.025 0.025 0) (0 0.025 0.025);
q1 = qua12 et qua34;
elim tol q1;
trac qual cach q1;
orie q1 dire (0 0 1);
*
pla7 =  q1 plus (0.0375 0.15   0);
pla8 =  q1 plus (0.10   0.15   0);
pla9 =  q1 plus (0.15   0.10   0);
depl pla9 tour 90 (0.175 0.125 0) (0.175 0.125 1);
pla10 = q1 plus (0.15   0.0375 0);
depl pla10 tour 90 (0.175 0.0625 0) (0.175 0.0625 1);
*
dens dd;
*con5 = p0 d (0.030 0 0) d (0.030 0.025 0) d (0.012 0.025 0)
*       c (0 0.025 0) (0 0.013 0) d p0;
con5 = p0 d (0.0375 0 0) d (0.0375 0.025 0) d (0.012 0.025 0)
       c 2 (0 0.025 0) (0 0.013 0) d p0;
trac qual cach con5;
qua5 = con5 surf plan;
trac qual cach qua5;
orie qua5 dire (0 0 1);
qua6 = qua5 syme plan (0 0.025 0) (0.030 0.025 0) (0 0.025 0.025);
orie qua6 dire (0 0 1);
q56 = qua5 et qua6;
elim tol q56;
trac qual cach q56;
orie q56 dire (0 0 1);
*
pla11 = q56 plus (0 0.15 0);
pla12 = q56 plus (0 0 0);
depl pla12 tour 90 (0 0.025 0) (0 0.025 0.025);
depl pla12 plus (0.175 -0.025 0);
*
*plate = pla1 et pla2 et pla3 et pla4 et pla5 et pla6 et
*        pla7 et pla8 et pla9 et pla10 et pla11 et pla12;
plate = pla1 et pla2 et pla3 et pla5 et
```

```
       pla7 et pla8 et pla9 et pla10 et pla11 et pla12;
elim tol plate;
orie plate dire (0 0 1);
trac qual cach plate;
*
*platet = plate elem tri3;
*orie platet dire (0 0 1);
plateq = plate elem qua4;
orie plateq dire (0 0 1);
*
* Lower frame
*
blf = (plate diff pla1) plus (0 0 -0.0154);
lframe = blf volu tran (0 0 0.015);
trac qual cach lframe;
lfra8 = lframe elem cub8;
*lfra6 = lframe elem pri6;
*
* Upper frame (without the bolts)
*
uf2 = pla2 plus (0 0 0.0004);
uf3 = pla3 plus (0 0 0.0004);
uf4 = pla4 plus (0 0 0.0004);
uf5 = pla5 plus (0 0 0.0004);
uf6 = pla6 plus (0 0 0.0004);
uf11 = pla11 plus (0 0 0.0004);
uf12 = pla12 plus (0 0 0.0004);
couf = p0 d (0.025 0 0) d (0.025 0.0190 0) c 2 (0.025 0.025 0)
       (0.0190 0.025 0) d (0 0.025 0) d p0;
quf1 = couf surf plan;
quf2 = quf1 syme plan (0.025 0 0) (0.025 0.025 0) (0.025 0 0.025);
quf12 = quf1 et quf2;
quf34 = quf12 syme plan (0 0.025 0) (0.025 0.025 0) (0 0.025 0.025);
u1 = quf12 et quf34;
elim tol u1;
orie u1 dire (0 0 1);
*
uf7 =  u1 plus (0.0375 0.15   0.0004);
uf8 =  u1 plus (0.10   0.15   0.0004);
uf9 =  u1 plus (0.15   0.10   0.0004);
uf10 = u1 plus (0.15   0.0375 0.0004);
*
*buf = uf2 et uf3 et uf4 et uf5 et uf6 et
*      uf7 et uf8 et uf9 et uf10 et uf11 et uf12;
buf = uf2 et uf3 et uf5 et
      uf7 et uf8 et uf9 et uf10 et uf11 et uf12;
elim tol buf;
uframe = buf volu tran (0 0 0.015);
trac qual cach uframe;
ufra8 = uframe elem cub8;
*ufra6 = uframe elem pri6;
*
* Bolts
*
*n = 4;
n = 2;
pz = 0 0 1;
b1 ier = px4cir3d (0.006 0 0) (0 0.006 0) p0 pz n tol;
b2 = b1 tour 90 p0 pz;
b3 = b2 tour 90 p0 pz;
b4 = b3 tour 90 p0 pz;
basu = b1 et b2 et b3 et b4;
elim tol basu;
basl = basu plus p0;
depl basu plus (0 0 0.0004);
bolu = basu volu tran 2 (0 0 0.0150);
boll = basl volu tran 2 (0 0 0.0158);
depl boll plus (0 0 -0.0154);
bolt = boll et bolu;
elim tol bolt;
trac qual cach bolt;
*
bolt1 = bolt plus (0.0625 0.1750 0);
bolt2 = bolt plus (0.1250 0.1750 0);
bolt3 = bolt plus (0.1750 0.1250 0);
bolt4 = bolt plus (0.1750 0.0625 0);
bolts = bolt1 et bolt2 et bolt3 et bolt4;
*
uframeb = uframe et bolts;
elim tol uframeb;
*
* Pressurized surfaces
*
pairb = pla1 coul noir;
ppimp = blf coul noir;
*ppimp3 = ppimp elem tri3;
ppimp4 = ppimp elem qua4;
trac qual cach (pairb et ppimp);
*
mesh = plate et lframe et uframeb et pairb et ppimp;
tass mesh noop;
sauv form mesh;
trac qual cach mesh;
*
fin;
```

a0bg02.epx

```
AOBG02
ECHO
!CONV WIN
```

```
CAST mesh
EROS 1.0
TRID LAGR
DIME ADAP NPOI 2000 Q4GS 2000 CL3D 2000 NPIN 2000 ENDA TERM
GEOM CUB8 lfra8 ufra8 bolts
     Q4GS plateq CL3D pairb ppimp4 TERM
COMP ORIE OBJE LECT plate TERM POIN 0 0 1
     EPAI 0.0008 LECT plate _q4gs TERM
     GROU 8 'pbol1'  LECT ppimp TERM
                     COND SPHE XC 0.0625 YC 0.175  ZC -0.0154 R 10.E-3
            'pbol2'  LECT ppimp TERM
                     COND SPHE XC 0.125  YC 0.175  ZC -0.0154 R 10.E-3
            'pbol3'  LECT ppimp TERM
                     COND SPHE XC 0.175  YC 0.125  ZC -0.0154 R 10.E-3
            'pbol4'  LECT ppimp TERM
                     COND SPHE XC 0.175  YC 0.0625 ZC -0.0154 R 10.E-3
            'pbolts' LECT pbol1 pbol2 pbol3 pbol4 TERM
            'PrCen'  LECT pairb TERM
                     COND NEAR POIN 0.0 0.0 0.0
            'PrSen6' LECT pairb TERM
                     COND NEAR POIN 0.175 0.0 0.0
            'pinada' LECT plate TERM
                     COND BOX XO 0.0 YO 0.0  ZO -0.1
                           DX 0.15 DY 0.15 DZ 0.2
     NGRO 9 'bloc'   LECT uframe TERM COND Z GT 0.01539
            'symx'   LECT mesh TERM COND X LT 0.0001
            'symy'   LECT mesh TERM COND Y LT 0.0001
            'symxr'  LECT plate TERM COND X LT 0.0001
            'symyr'  LECT plate TERM COND Y LT 0.0001
            'cen'    LECT plate TERM COND NEAR POIN 0.0 0.0 0.0
            'axis1'  LECT plate TERM COND X LT 0.0001
            'axis2'  LECT plate TERM COND Y LT 0.0001
            'plateN' LECT plate DIFF axis1 axis2 TERM
     COUL VERT LECT plate  TERM
          ROSE LECT lframe TERM
          TURQ LECT uframe bolts TERM
          ROUG LECT pairb  TERM
          JAUN LECT pbolts TERM
          GR50 LECT ppimp DIFF pbolts TERM
ADAP THRS ECRO 11 TMIN 0.1 TMAX 0.5 MAXL 3
          LECT pinada TERM
MATE VPJC RO 2700.0 YOUN 70.0E9 NU 0.3 ELAS 80.0E6 mxit 500
          QR1 49.3E6 CR1 1457.1 QR2 5.2E6 CR2 121.5
          PDOT 5.E-4 C 0.014 TQ 0.9 CP 910.0
          TM 893.0 M 0.0 DC 1.0 WC 44.6E6
     LECT plate _q4gs TERM
     VPJC RO 7850.0 YOUN 2.1E11 NU 0.33 ELAS 3.7E8 mxit 500
          QR1 2.364E8 CR1 39.3 QR2 4.081E8 CR2 4.5
          PDOT 5.E-4 C 1.E-3 TQ 0.9 CP 452.0
          TM 1800.0 M 0.0 DC 0.9 WC 473.0E6
          LECT lframe uframe bolts TERM
     IMPE PIMP RO 7850.0 PRES 0 PREF 0
          LECT ppimp DIFF pbol1 pbol2 pbol3 pbol4 TERM
     IMPE PIMP RO 7850.0 PRES 340.e6 PREF 0
          LECT pbol1 pbol2 pbol3 pbol4 TERM
     IMPE AIRB X 0.0 Y 0.0 Z -0.25
          MASS 0.0402 TAUT ANGL COEF 1.0
          LECT pairb _cl3d TERM
LINK COUP SPLT NONE SOLV PARD
     BLOQ 123 LECT bloc TERM
     BLOQ 1 LECT symx TERM
     BLOQ 2 LECT symy TERM
     BLOQ 56 LECT symxr TERM
     BLOQ 46 LECT symyr TERM
     PINB SELF MLEV 0 LECT pinada TERM
     GLIS 2 FROT MUST 0.1 MUDY 0.1 GAMM 0 ! Contact surface #1
            PGAP 0.4E-3
            MAIT LECT lframe TERM
            PESC LECT plateN  TERM
            FROT MUST 0.1 MUDY 0.1 GAMM 0 ! Contact surface #2
            PGAP 0.4E-3
            MAIT LECT uframeb TERM
            PESC LECT plateN   TERM
ECRI VITE TFRE 1.E-3
          POIN LECT 1 TERM
          NOEL
     FICH SPLI ALIC TFRE 1.0E-4
OPTI NOTE
     JAUM
     CSTA 0.5
     GLIS NORM ELEM
     LOG 1
     LMST
     ADAP RCON
     PINS GRID DPIN 1.01
MEAS ! Measurement commands (batch)
  OBJE LECT pbolts TERM
  EMIN LECT pbolts TERM
  EMAX LECT pbolts TERM
CALC TINI 0.0 TEND 10.E-3
FIN
```

a0bg02a.epx

```
AOBG02A
ECHO
 CONV WIN
OPTI PRIN
RESU SPLI ALIC 'a0bg02.ali' GARD PSCR
SORT VISU NSTO 1
*=====================================================================
```

```
PLAY
CAME   1 EYE  -7.24266E-01 -6.53429E-01  1.00071E+00 ! Tilted view
!        Q     8.72148E-01  3.06624E-01 -3.77772E-01  5.12661E-02
         VIEW  6.27507E-01  5.73576E-01 -5.26541E-01
         RIGH  7.09321E-01 -1.42244E-01  6.90385E-01
         UP   -3.21091E-01  8.06707E-01  4.96109E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  9.95290E-02  9.95654E-02  3.09460E-01
!RSPHERE:  3.54812E-01
!RADIUS :  1.31281E+00
!ASPECT :  1.00000E+00
!NEAR   :  9.57994E-01
!FAR    :  2.02243E+00
CAME   2 EYE   9.95290E-02  9.95654E-02 -5.77571E-01! View from back
!        Q    -2.05103E-10  0.00000E+00 -1.00000E+00  0.00000E+00
         VIEW -4.10207E-10  0.00000E+00  1.00000E+00
         RIGH -1.00000E+00  0.00000E+00 -4.10207E-10
         UP    0.00000E+00  1.00000E+00  0.00000E+00
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  9.95290E-02  9.95654E-02  3.09460E-01
!RSPHERE:  3.54812E-01
!RADIUS :  8.87031E-01
!ASPECT :  1.00000E+00
!NEAR   :  5.32219E-01
!FAR    :  1.59666E+00
SCEN GEOM NAVI FREE
          FACE SBAC
    !     PINB CDES JOIN
    !VECT SCCO FIEL VITE SCAL USER PROG 0 PAS 10 70 TERM
    !ISO  FELE FIEL ECRO 3 SCAL USER PROG 0.02 PAS 0.02 0.28 TERM
     LIMA ON
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 101 FPS 15 KFRE 10 COMP -1
     OBJE LECT tous DIFF pairb TERM NFAI REND
GOTR LOOP 99 OFFS FICH AVI CONT NOCL
     OBJE LECT tous DIFF pairb TERM NFAI REND
GO
TRAC OFFS FICH AVI CONT
     OBJE LECT tous DIFF pairb TERM NFAI REND
ENDPLAY
*=====================================================================
FIN
```

a0bg02z.epx

```
AOBG02Z
ECHO
 CONV WIN
OPTI PRIN
RESU SPLI ALIC 'a0bg02.ali' GARD PSCR
SORT VISU NSTO 101
FIN
```

adpi04.epx

```
ADPI04
ECHO
 CONV WIN
LAGR CPLA
DIME ADAP NPOI 62 Q42L 56 NPIN 56 ENDA TERM
GEOM LIBR POIN 8 Q42L 2 TERM
  0 0.415  1 0.415  1 1.415  0 1.415
  0 -1  1 -1  1 0  0 0
 1 2 3 4
 5 6 7 8
COMP EPAI 1. LECT 1 2 _q42l TERM
MATE VM23 RO 8000. YOUN 1.D11 NU 0.3 ELAS 2.D8
          TRAC 3 2.D8 2.D-3 3.D8 1. 3.1D8 2.
          LECT 1 2 _q42l TERM
LINK COUP
     PINB BODY LECT 1 TERM
          BODY LECT 2 TERM
INIT VITE 2 -100 LECT 1 PAS 1 4 TERM
     VITE 2  100 LECT 5 PAS 1 8 TERM
ECRI DEPL VITE ACCE FINT FEXT FLIA FDEC CONT ECRO TFRE 1.E-3
     FICH ALIC FREQ 1
OPTI PAS AUTO NOTE LOG 1
     PINS DUMP STAT
CALC TINI 0. TEND 0.0048D0
PLAY
CAME   1 EYE   5.00000E-01  2.60000E-02  5.71296E+00
!        Q     1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
         VIEW  0.00000E+00  0.00000E+00 -1.00000E+00
         RIGH  1.00000E+00  0.00000E+00  0.00000E+00
         UP    0.00000E+00  1.00000E+00  0.00000E+00
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E-01  2.60000E-02  0.00000E+00
!RSPHERE:  1.32860E+00
!RADIUS :  5.71296E+00
!ASPECT :  1.00000E+00
!NEAR   :  4.38437E+00
!FAR    :  8.37016E+00
SCEN GEOM NAVI FREE
          PINB JOIN
     VECT SCCO FIEL VITE SCAL USER PROG 0 PAS 10 70 TERM
     ISO  FELE FIEL ECRO 3 SCAL USER PROG 0.02 PAS 0.02 0.28 TERM
SLER CAM1 1 NFRA 1
ADAP
  SPLI 1
  SPLI 2
  SPLI 3
  SPLI 4
  SPLI 9
  SPLI 10
  SPLI 11
  SPLI 12
  SPLI 15
  SPLI 16
  SPLI 21
  SPLI 22
  SPLI 25
  SPLI 26
TERM
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 200 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 198 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
FIN
```

adpi05.epx

```
ADPI05
ECHO
 CONV WIN
LAGR CPLA
DIME ADAP NPOI 62 Q42L 56 NPIN 56 ENDA TERM
GEOM LIBR POIN 8 Q42L 2 TERM
  0 0.052  1 0.052  1 1.052  0 1.052
  0 -1  1 -1  1 0  0 0
 1 2 3 4
 5 6 7 8
COMP EPAI 1. LECT 1 2 _q42l TERM
MATE VM23 RO 8000. YOUN 1.D11 NU 0.3 ELAS 2.D8
          TRAC 3 2.D8 2.D-3 3.D8 1. 3.1D8 2.
          LECT 1 2 _q42l TERM
LINK COUP
     PINB BODY LECT 1 TERM
          BODY LECT 2 TERM
INIT VITE 2 -100 LECT 1 PAS 1 4 TERM
     VITE 2  100 LECT 5 PAS 1 8 TERM
     ADAP SPLI LEVE 2 LECT 1 2 TERM
          SPLI LEVE 3 LECT 3 4 9 10 TERM
          SPLI LEVE 4 LECT 11 12 15 16 21 22 25 26 TERM
ECRI DEPL VITE ACCE FINT FEXT FLIA FDEC CONT ECRO TFRE 1.E-3
     FICH ALIC FREQ 1
OPTI PAS AUTO NOTE LOG 1
     PINS DUMP STAT
CALC TINI 0. TEND 0.003D0
PLAY
CAME   1 EYE   5.00000E-01  2.60000E-02  5.71296E+00
!        Q     1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
         VIEW  0.00000E+00  0.00000E+00 -1.00000E+00
         RIGH  1.00000E+00  0.00000E+00  0.00000E+00
         UP    0.00000E+00  1.00000E+00  0.00000E+00
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E-01  2.60000E-02  0.00000E+00
!RSPHERE:  1.32860E+00
!RADIUS :  5.71296E+00
!ASPECT :  1.00000E+00
!NEAR   :  4.38437E+00
!FAR    :  8.37016E+00
SCEN GEOM NAVI FREE
          PINB JOIN
     VECT SCCO FIEL VITE SCAL USER PROG 0 PAS 10 70 TERM
     ISO  FELE FIEL ECRO 3 SCAL USER PROG 0.02 PAS 0.02 0.28 TERM
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 141 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 139 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
FIN
```

bm_mpi_adap_pinballs.epx

```
$ BM_MPI_ADAP_PINBALLS       TOUS VFAUCHER  15/01/21 21:41:02 #2816
Test MPI 2D adap + pinballs

ECHO BMPI

DPLA SCLM ROB DACT 1
ADAP MLVL 4

CASTEM TOUT

DIME TERM

GEOM CAR1 STR1 STR2 TERM

ADAP
!    Criterion based on the velocity curvature
     INDI VITE
          EMIN 1.E-3
```

```
     TYPE CURV
     STRA PELE 5000 ALFA 4
     CERR 1.0 LECT STR1 STR2 TERM

     PCLD
         GAP MASTER LECT STR1 TERM
             SLAVE  LECT STR2 TERM
             MAXL 3
             RADI 0.06

         GAP MASTER LECT STR2 TERM
             SLAVE  LECT STR1 TERM
             MAXL 3
             RADI 0.06

          INDI NIND 1
               ELEM LECT STR1 STR2 TERM
               MAXL 3

MATE
    LINE YOUN 210.E7 NU 0.3 RO 7800. LECT STR1 STR2 TERM

LINK COUP
    BLOQ 12 LECT LBL1 TERM
    PINBALL BODY LECT STR1 TERM
            BODY LECT STR2 TERM

INIT VITE 2 -10. LECT STR2 TERM

ECRI COOR DEPL VITE ACCE FINT FEXT CONT ECRO FREQ 10
     NOPO NOEL
     FICH FORM PVTK TFREQ 4.E-4
          MPI
          GROUP AUTO
          VARI DEPL VITE ACCE FEXT FINT ECRO INDI
          PCLD
          PINB


OPTI NOTE STEP IO CSTA 0.5
     PINS EQVL
     ADAP STAT
          MAXL 4
          PCLD MODE 2

CALC TINI 0. TEND 2.E-2

QUALIFICATION
   DEPL COMP 1 LECT 232 TERM REFE -2.28660E-02 TOLE 5.E-2
   DEPL COMP 2 LECT 232 TERM REFE -4.23665E-02 TOLE 1.E-2

FIN

BEGIN DESCRIPTION
Bench for 2D-adaptivity contact by pinballs with MPI.
Impact and rebound  of a solid on another.
Combined adaptivity criteria near the contact surface and linked to the
curvature of the velocity field.
Qualification on the displacement of the bottom left corner of the
impacting solid.
END DESCRIPTION
```

piad01.epx

```
PIAD01
ECHO
 CONV WIN
LAGR CPLA
DIME ADAP NPOI 92 Q42L 88 NPIN 88 ENDA TERM
GEOM LIBR POIN 24 Q42L 11 TERM
  0 0  1 0  2 0  3 0  4 0  5 0
  0 1  1 1  2 1  3 1  4 1  5 1
       1 2  2 2  3 2  4 2
       1 3  2 3  3 3  4 3
       1 4  2 4  3 4  4 4
 1 2 8 7
 2 3 9 8
 3 4 10 9
 4 5 11 10
 5 6 12 11
 13 14 18 17
 14 15 19 18
 15 16 20 19
 17 18 22 21
 18 19 23 22
 19 20 24 23
COMP GROU 2 'target' LECT 1 PAS 1 5 TERM
            'projec' LECT 6 PAS 1 11 TERM
     EPAI 1.0 LECT target projec _q42l TERM
MATE VM23 RO 8000. YOUN 1.D11 NU 0.3 ELAS 1.D11
          TRAC 1 1.D11 1.0
          LECT target projec _q42l TERM
OPTI PINS EQVL
LINK COUP SPLT NONE
     PINB BODY MLEV 0 LECT target TERM
          BODY MLEV 0 LECT projec TERM
     ADAP LMAX 3 SCAL 1.0 NOUN
INIT VITE 2 -100 LECT projec TERM
ECRI DEPL VITE ACCE FINT FEXT FLIA FDEC CONT ECRO TFRE 0.005
     FICH ALIC TFRE 0.0005
OPTI PAS AUTO NOTE LOG 1
     CSTA 0.5
CALC TINI 0. TEND 0.06D0
*=================================================================
PLAY
CAME   1 EYE   2.50000E+00  2.02417E-01  2.11668E+01
!        Q     1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
         VIEW  0.00000E+00  0.00000E+00 -1.00000E+00
         RIGH  1.00000E+00  0.00000E+00  0.00000E+00
         UP    0.00000E+00  1.00000E+00  0.00000E+00
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  2.50000E+00  2.02417E-01  0.00000E+00
!RSPHERE:  4.60149E+00
!RADIUS :  2.11668E+01
!ASPECT :  1.00000E+00
!NEAR   :  1.65654E+01
!FAR    :  3.03698E+01
SCEN GEOM NAVI FREE
          FACE HFRO
          PINB PARE
     VECT SCCO FIEL VITE SCAL USER PROG 10 PAS 10 140 TERM
SLER CAM1 1 NFRA 1
FREQ 0 TFRE 0.5E-3
TRAC OFFS FICH AVI NOCL NFTO 121 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 119 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
*=================================================================
FIN
```

piad02.epx

```
PIAD02
ECHO
 CONV WIN
LAGR CPLA
DIME ADAP NPOI 92 Q42L 88 NPIN 88 ENDA TERM
GEOM LIBR POIN 24 Q42L 11 TERM
  0 0  1 0  2 0  3 0  4 0  5 0
  0 1  1 1  2 1  3 1  4 1  5 1
       1 2  2 2  3 2  4 2
       1 3  2 3  3 3  4 3
       1 4  2 4  3 4  4 4
 1 2 8 7
 2 3 9 8
 3 4 10 9
 4 5 11 10
 5 6 12 11
 13 14 18 17
 14 15 19 18
 15 16 20 19
 17 18 22 21
 18 19 23 22
 19 20 24 23
COMP GROU 2 'target' LECT 1 PAS 1 5 TERM
            'projec' LECT 6 PAS 1 11 TERM
     EPAI 1.0 LECT target projec _q42l TERM
MATE VM23 RO 8000. YOUN 1.D11 NU 0.3 ELAS 1.D11
          TRAC 1 1.D11 1.0
          LECT target projec _q42l TERM
OPTI PINS EQVL
LINK COUP SPLT NONE
     PINB BODY MLEV 0 LECT target TERM
          BODY MLEV 0 LECT projec TERM
     ADAP LMAX 3 SCAL 1.0
INIT VITE 2 -100 LECT projec TERM
ECRI DEPL VITE ACCE FINT FEXT FLIA FDEC CONT ECRO TFRE 0.005
     FICH ALIC TFRE 0.0005
OPTI PAS AUTO NOTE LOG 1
     CSTA 0.5
CALC TINI 0. TEND 0.06D0
*=================================================================
PLAY
CAME   1 EYE   2.50000E+00  2.02417E-01  2.11668E+01
!        Q     1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
         VIEW  0.00000E+00  0.00000E+00 -1.00000E+00
         RIGH  1.00000E+00  0.00000E+00  0.00000E+00
         UP    0.00000E+00  1.00000E+00  0.00000E+00
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  2.50000E+00  2.02417E-01  0.00000E+00
!RSPHERE:  4.60149E+00
!RADIUS :  2.11668E+01
!ASPECT :  1.00000E+00
!NEAR   :  1.65654E+01
!FAR    :  3.03698E+01
SCEN GEOM NAVI FREE
          FACE HFRO
          PINB PARE
     VECT SCCO FIEL VITE SCAL USER PROG 10 PAS 10 140 TERM
SLER CAM1 1 NFRA 1
FREQ 0 TFRE 0.5E-3
TRAC OFFS FICH AVI NOCL NFTO 121 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 119 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
*=================================================================
FIN
```

piad03.epx

```
PIAD03
ECHO
 CONV WIN
LAGR CPLA
GEOM LIBR POIN 24 Q42L 11 TERM
  0 0  1 0  2 0  3 0  4 0  5 0
  0 1  1 1  2 1  3 1  4 1  5 1
       1 2  2 2  3 2  4 2
       1 3  2 3  3 3  4 3
       1 4  2 4  3 4  4 4
 1 2 8 7
 2 3 9 8
 3 4 10 9
 4 5 11 10
 5 6 12 11
 13 14 18 17
 14 15 19 18
 15 16 20 19
 17 18 22 21
 18 19 23 22
 19 20 24 23
COMP GROU 2 'target' LECT 1 PAS 1 5 TERM
           'projec' LECT 6 PAS 1 11 TERM
     EPAI 1.0 LECT target projec TERM
MATE VM23 RO 8000. YOUN 1.D11 NU 0.3 ELAS 1.D11
          TRAC 1 1.D11 1.0
          LECT target projec TERM
OPTI PINS EQVL
LINK COUP SPLT NONE
     PINB BODY MLEV 2 LECT target TERM
          BODY MLEV 2 LECT projec TERM
INIT VITE 2 -100 LECT projec TERM
ECRI DEPL VITE ACCE FINT FEXT FLIA FDEC CONT ECRO TFRE 0.005
     FICH ALIC TFRE 0.0005
OPTI PAS AUTO NOTE LOG 1
     CSTA 0.5
CALC TINI 0. TEND 0.06D0
*=====================================================================
PLAY
CAME   1 EYE   2.50000E+00  2.02417E-01  2.11668E+01
!        Q     1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
         VIEW  0.00000E+00  0.00000E+00 -1.00000E+00
         RIGH  1.00000E+00  0.00000E+00  0.00000E+00
         UP    0.00000E+00  1.00000E+00  0.00000E+00
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  2.50000E+00  2.02417E-01  0.00000E+00
!RSPHERE:  4.60149E+00
!RADIUS :  2.11668E+01
!ASPECT :  1.00000E+00
!NEAR   :  1.65654E+01
!FAR    :  3.03698E+01
SCEN GEOM NAVI FREE
          FACE HFRO
          PINB PARE CDES
     VECT SCCO FIEL VITE SCAL USER PROG 10 PAS 10 140 TERM
SLER CAM1 1 NFRA 1
FREQ 0 TFRE 0.5E-3
TRAC OFFS FICH AVI NOCL NFTO 121 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 119 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
*=====================================================================
FIN
```

piad04.epx

```
PIAD04
ECHO
 CONV WIN
LAGR CPLA
DIME ADAP NPOI 92 Q42L 88 NPIN 88 ENDA TERM
GEOM LIBR POIN 24 Q42L 11 TERM
  0 0  1 0  2 0  3 0  4 0  5 0
  0 1  1 1  2 1  3 1  4 1  5 1
       1 2  2 2  3 2  4 2
       1 3  2 3  3 3  4 3
       1 4  2 4  3 4  4 4
 1 2 8 7
 2 3 9 8
 3 4 10 9
 4 5 11 10
 5 6 12 11
 13 14 18 17
 14 15 19 18
 15 16 20 19
 17 18 22 21
 18 19 23 22
 19 20 24 23
COMP GROU 2 'target' LECT 1 PAS 1 5 TERM
           'projec' LECT 6 PAS 1 11 TERM
     EPAI 1.0 LECT target projec _q42l TERM
MATE VM23 RO 8000. YOUN 1.D11 NU 0.3 ELAS 1.D11
          TRAC 1 1.D11 1.0
          LECT target projec _q42l TERM
OPTI PINS EQVL
LINK COUP SPLT NONE
     PINB BODY MLEV 3 LECT target TERM
          BODY MLEV 3 LECT projec TERM
     ADAP LMAX 3 SCAL 1.0
INIT VITE 2 -100 LECT projec TERM
ECRI DEPL VITE ACCE FINT FEXT FLIA FDEC CONT ECRO TFRE 0.005
     FICH ALIC TFRE 0.0005
OPTI PAS AUTO NOTE LOG 1
     CSTA 0.5
CALC TINI 0. TEND 0.06D0
*=====================================================================
PLAY
CAME   1 EYE   2.50000E+00  2.02417E-01  2.11668E+01
!        Q     1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
         VIEW  0.00000E+00  0.00000E+00 -1.00000E+00
         RIGH  1.00000E+00  0.00000E+00  0.00000E+00
         UP    0.00000E+00  1.00000E+00  0.00000E+00
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  2.50000E+00  2.02417E-01  0.00000E+00
!RSPHERE:  4.60149E+00
!RADIUS :  2.11668E+01
!ASPECT :  1.00000E+00
!NEAR   :  1.65654E+01
!FAR    :  3.03698E+01
SCEN GEOM NAVI FREE
          FACE HFRO
          PINB PARE CDES
     VECT SCCO FIEL VITE SCAL USER PROG 10 PAS 10 140 TERM
SLER CAM1 1 NFRA 1
FREQ 0 TFRE 0.5E-3
TRAC OFFS FICH AVI NOCL NFTO 121 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 119 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
*=====================================================================
FIN
```

pith07.dgibi

```
opti echo 1;
opti titr 'pith07';
opti dime 3 elem cub8;
opti trac psc ftra 'pith07_mesh.ps';
opti sauv form 'pith07.msh';
*
p1 = -3 -3 0;
p2 =  3 -3 0;
p3 =  3  0 0;
p4 = -3  0 0;
c1 = p1 d 12 p2;
c2 = p2 d  6 p3;
c3 = p3 d 12 p4;
c4 = p4 d  6 p1;
bastar = dall c1 c2 c3 c4 plan;
target = bastar volu tran 1 (0 0 0.5);
*
gap = 0.1;
tol = 0.01;
*
p5 = -5  (5 + gap) 0;
p6 =  0  (0 + gap) 0;
p7 =  5  (5 + gap) 0;
p8 =  4  (5 + gap) 0;
p9 =  0  (1 + gap) 0;
p10= -4  (5 + gap) 0;
pc =  0  (5 + gap) 0;
c5 = p10 d 2 p5;
proj1 = c5 rota 16 90 pc (pc plus (0 0 1));
c6 = p9  d 2 p6;
proj2 = c6 rota 16 90 pc (pc plus (0 0 1));
baspro = proj1 et proj2;
elim tol (baspro et p7 et p8);
projec = baspro volu tran 1 (0 0 0.5);
*
mesh = target et projec;
*
tass mesh noop;
trac cach qual mesh;
*
sauv form mesh;
fin;
```

pith07.epx

```
PITH07
ECHO
!CONV WIN
EROS 1.0
CAST mesh
LAGR TRID
DIME ADAP NPOI 2993 CUBE 2152 NPIN 2152 ENDA TERM
GEOM CUBE target projec TERM
COMP COUL VERT LECT target TERM
          TURQ LECT projec TERM
     NGRO 1 'bloc' LECT target TERM COND Y LT -2.99
ADAP THRS ECRO 13 TMIN 0.1 TMAX 0.5 MAXL 3
          LECT projec TERM
MATE LINE RO 8000. YOUN 1.D11 NU 0.3
          LECT target TERM
DPDC RO 2360 YOUN 40.1E+9 NU 0.2
     FC 75.E+6 DAGG 16.0E-3  VERS 8 EFVI
```

```
     EROD ENDT 0.99 ENDC 0.99 DVOL 0.3
          LECT projec _cube TERM
OPTI PINS EQVL
LINK COUP SPLT NONE
     BLOQ 123 LECT bloc TERM
     PINB BODY MLEV 0 LECT target TERM
          SELF MLEV 0 LECT projec TERM
INIT VITE 2 -100 LECT projec TERM
ECRI DEPL VITE ACCE FINT FEXT FLIA FDEC CONT ECRO TFRE 0.005
     FICH ALIC TFRE 0.0005
OPTI PAS AUTO NOTE LOG 1
     CSTA 0.5
     ADAP RCON
     PINS GRID DPIN 1.01
CALC TINI 0. TEND 0.02D0
FIN
```

pith07a.epx

```
PITH07A
ECHO
 CONV WIN
OPTI PRIN
RESU ALIC 'pith07.ali' GARD PSCR
SORT VISU NSTO 1
*=====================================================================
PLAY
CAME   1 EYE   0.00000E+00  1.01980E+00  2.61623E+01 ! Front view
!        Q     1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
         VIEW  0.00000E+00  0.00000E+00 -1.00000E+00
         RIGH  1.00000E+00  0.00000E+00  0.00000E+00
         UP    0.00000E+00  1.00000E+00  0.00000E+00
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  0.00000E+00  1.01980E+00  2.50000E-01
!RSPHERE:  6.47808E+00
!RADIUS :  2.59123E+01
!ASPECT :  1.00000E+00
!NEAR   :  1.94342E+01
!FAR    :  3.88685E+01
CAME   2 EYE  -1.56209E+01  9.23770E+00  1.44814E+01 ! Tilted view
!        Q     9.00472E-01 -1.77386E-01 -3.93317E-01 -5.46453E-02
         VIEW  6.88956E-01 -3.62449E-01 -6.27671E-01
         RIGH  6.84631E-01  4.11253E-02  7.27729E-01
         UP    2.37951E-01  9.31096E-01 -2.76477E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  0.00000E+00  1.01980E+00  2.50000E-01
!RSPHERE:  6.47808E+00
!RADIUS :  2.26733E+01
!ASPECT :  1.00000E+00
!NEAR   :  1.61952E+01
!FAR    :  3.56294E+01
SCEN GEOM NAVI FREE
          !FACE HFRO
          !PINB PARE
     !VECT SCCO FIEL VITE SCAL USER PROG 10 PAS 10 140 TERM
      LIMA ON
SLER CAM1 2 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 41 FPS 15 KFRE 10 COMP -1 NFAI REND
GOTR LOOP 39 OFFS FICH AVI CONT NOCL NFAI REND
GO
TRAC OFFS FICH AVI CONT NFAI REND
ENDPLAY
*=====================================================================
FIN
```

pith07b.epx

```
PITH07B
ECHO
 CONV WIN
OPTI PRIN
RESU ALIC 'pith07.ali' GARD PSCR
SORT VISU NSTO 1
*=====================================================================
PLAY
CAME   1 EYE   0.00000E+00  1.01980E+00  2.61623E+01 ! Front view
!        Q     1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
         VIEW  0.00000E+00  0.00000E+00 -1.00000E+00
         RIGH  1.00000E+00  0.00000E+00  0.00000E+00
         UP    0.00000E+00  1.00000E+00  0.00000E+00
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  0.00000E+00  1.01980E+00  2.50000E-01
!RSPHERE:  6.47808E+00
!RADIUS :  2.59123E+01
!ASPECT :  1.00000E+00
!NEAR   :  1.94342E+01
!FAR    :  3.88685E+01
CAME   2 EYE  -1.56209E+01  9.23770E+00  1.44814E+01 ! Tilted view
!        Q     9.00472E-01 -1.77386E-01 -3.93317E-01 -5.46453E-02
         VIEW  6.88956E-01 -3.62449E-01 -6.27671E-01
         RIGH  6.84631E-01  4.11253E-02  7.27729E-01
         UP    2.37951E-01  9.31096E-01 -2.76477E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  0.00000E+00  1.01980E+00  2.50000E-01
!RSPHERE:  6.47808E+00
!RADIUS :  2.26733E+01
!ASPECT :  1.00000E+00
!NEAR   :  1.61952E+01
!FAR    :  3.56294E+01
SCEN GEOM NAVI FREE
          FACE HFRO
         !LINE HEOU SSHA
          PINB PARE CDES
    !VECT SCCO FIEL VITE SCAL USER PROG 10 PAS 10 140 TERM
     LIMA ON
SLER CAM1 2 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 41 FPS 15 KFRE 10 COMP -1 NFAI REND
GOTR LOOP 39 OFFS FICH AVI CONT NOCL NFAI REND
GO
TRAC OFFS FICH AVI CONT NFAI REND
ENDPLAY
*=====================================================================
FIN
```

pith07c.epx

```
PITH07C
ECHO
 CONV WIN
OPTI PRIN
RESU ALIC 'pith07.ali' GARD PSCR
SORT VISU NSTO 1
*=====================================================================
PLAY
CAME   1 EYE   0.00000E+00  1.01980E+00  2.61623E+01 ! Front view
!        Q     1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
         VIEW  0.00000E+00  0.00000E+00 -1.00000E+00
         RIGH  1.00000E+00  0.00000E+00  0.00000E+00
         UP    0.00000E+00  1.00000E+00  0.00000E+00
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  0.00000E+00  1.01980E+00  2.50000E-01
!RSPHERE:  6.47808E+00
!RADIUS :  2.59123E+01
!ASPECT :  1.00000E+00
!NEAR   :  1.94342E+01
!FAR    :  3.88685E+01
CAME   2 EYE  -1.56209E+01  9.23770E+00  1.44814E+01 ! Tilted view
!        Q     9.00472E-01 -1.77386E-01 -3.93317E-01 -5.46453E-02
         VIEW  6.88956E-01 -3.62449E-01 -6.27671E-01
         RIGH  6.84631E-01  4.11253E-02  7.27729E-01
         UP    2.37951E-01  9.31096E-01 -2.76477E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  0.00000E+00  1.01980E+00  2.50000E-01
!RSPHERE:  6.47808E+00
!RADIUS :  2.26733E+01
!ASPECT :  1.00000E+00
!NEAR   :  1.61952E+01
!FAR    :  3.56294E+01
SCEN GEOM NAVI FREE
          FACE HFRO
         !LINE HEOU SSHA
          PINB CDES
    !VECT SCCO FIEL VITE SCAL USER PROG 10 PAS 10 140 TERM
     LIMA ON
SLER CAM1 2 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 41 FPS 15 KFRE 10 COMP -1 NFAI REND
GOTR LOOP 39 OFFS FICH AVI CONT NOCL NFAI REND
GO
TRAC OFFS FICH AVI CONT NFAI REND
ENDPLAY
*=====================================================================
FIN
```

pith07d.epx

```
PITH07D
ECHO
 CONV WIN
OPTI PRIN
RESU ALIC 'pith07.ali' GARD PSCR
SORT VISU NSTO 1
*=====================================================================
PLAY
CAME   1 EYE   0.00000E+00  1.01980E+00  2.61623E+01 ! Front view
!        Q     1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
         VIEW  0.00000E+00  0.00000E+00 -1.00000E+00
         RIGH  1.00000E+00  0.00000E+00  0.00000E+00
         UP    0.00000E+00  1.00000E+00  0.00000E+00
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  0.00000E+00  1.01980E+00  2.50000E-01
!RSPHERE:  6.47808E+00
!RADIUS :  2.59123E+01
!ASPECT :  1.00000E+00
!NEAR   :  1.94342E+01
!FAR    :  3.88685E+01
CAME   2 EYE  -1.56209E+01  9.23770E+00  1.44814E+01 ! Tilted view
!        Q     9.00472E-01 -1.77386E-01 -3.93317E-01 -5.46453E-02
         VIEW  6.88956E-01 -3.62449E-01 -6.27671E-01
         RIGH  6.84631E-01  4.11253E-02  7.27729E-01
         UP    2.37951E-01  9.31096E-01 -2.76477E-01
         FOV   2.48819E+01
```

```
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  0.00000E+00  1.01980E+00  2.50000E-01
!RSPHERE:  6.47808E+00
!RADIUS :  2.26733E+01
!ASPECT :  1.00000E+00
!NEAR   :  1.61952E+01
!FAR    :  3.56294E+01
SCEN GEOM NAVI FREE
          FACE HFRO
         !LINE HEOU SSHA
          PINB PARE
    !VECT SCCO FIEL VITE SCAL USER PROG 10 PAS 10 140 TERM
     LIMA ON
SLER CAM1 2 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 41 FPS 15 KFRE 10 COMP -1 NFAI REND
GOTR LOOP 39 OFFS FICH AVI CONT NOCL NFAI REND
GO
TRAC OFFS FICH AVI CONT NFAI REND
ENDPLAY
*=====================================================================
FIN
```

pith07z.epx

```
PITH07Z
ECHO
 CONV WIN
OPTI PRIN
RESU ALIC 'pith07.ali' GARD PSCR
SORT VISU NSTO 41
FIN
```

pith08.dgibi

```
opti echo 1;
opti titr 'pith08';
opti dime 3 elem cub8;
opti trac psc ftra 'pith08_mesh.ps';
opti sauv form 'pith08.msh';
*
p1 = -3 -3 0;
p2 =  3 -3 0;
p3 =  3  0 0;
p4 = -3  0 0;
c1 = p1 d 12 p2;
c2 = p2 d  6 p3;
c3 = p3 d 12 p4;
c4 = p4 d  6 p1;
bastar = dall c1 c2 c3 c4 plan;
target = bastar volu tran 1 (0 0 0.5);
*
gap = 0.1;
tol = 0.01;
*
p5 = -5  (5 + gap) 0;
p6 =  0  (0 + gap) 0;
p7 =  5  (5 + gap) 0;
p8 =  4  (5 + gap) 0;
p9 =  0  (1 + gap) 0;
p10= -4  (5 + gap) 0;
pc =  0  (5 + gap) 0;
c5 = p10 d 2 p5;
proj1 = c5 rota 16 90 pc (pc plus (0 0 1));
c6 = p9  d 2 p6;
proj2 = c6 rota 16 90 pc (pc plus (0 0 1));
baspro = proj1 et proj2;
elim tol (baspro et p7 et p8);
projec = baspro volu tran 1 (0 0 0.5);
*
mesh = target et projec;
*
tass mesh noop;
trac cach qual mesh;
*
sauv form mesh;
fin;
```

pith08.epx

```
PITH08
ECHO
!CONV WIN
EROS 1.0
CAST mesh
LAGR TRID
DIME ADAP NPOI 6714 CUBE 5368 NPIN 5368 ENDA TERM
GEOM CUBE target projec TERM
COMP COUL VERT LECT target TERM
          TURQ LECT projec TERM
     NGRO 1 'bloc' LECT target TERM COND Y LT -2.99
MATE LINE RO 8000. YOUN 1.D11 NU 0.3
          LECT target TERM
DPDC RO 2360 YOUN 40.1E+9 NU 0.2
     FC 75.E+6 DAGG 16.0E-3  VERS 8 EFVI
     EROD ENDT 0.99 ENDC 0.99 DVOL 0.3
          LECT projec _cube TERM
OPTI PINS EQVL
LINK COUP SPLT NONE
     BLOQ 123 LECT bloc TERM
     PINB BODY MLEV 0 LECT target TERM
          SELF MLEV 0 LECT projec TERM
          ADAP LMAX 3 SCAL 1.0 SCAS 0.55
INIT VITE 2 -100 LECT projec TERM
ECRI DEPL VITE ACCE FINT FEXT FLIA FDEC CONT ECRO TFRE 0.005
     FICH ALIC TFRE 0.0005
OPTI PAS AUTO NOTE LOG 1
     CSTA 0.5
     ADAP RCON
     PINS GRID DPIN 1.01
CALC TINI 0. TEND 0.02D0
FIN
```

pith08a.epx

```
PITH08A
ECHO
 CONV WIN
OPTI PRIN
RESU ALIC 'pith08.ali' GARD PSCR
SORT VISU NSTO 1
*=====================================================================
PLAY
CAME   1 EYE   0.00000E+00  1.01980E+00  2.61623E+01 ! Front view
!        Q     1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
         VIEW  0.00000E+00  0.00000E+00 -1.00000E+00
         RIGH  1.00000E+00  0.00000E+00  0.00000E+00
         UP    0.00000E+00  1.00000E+00  0.00000E+00
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  0.00000E+00  1.01980E+00  2.50000E-01
!RSPHERE:  6.47808E+00
!RADIUS :  2.59123E+01
!ASPECT :  1.00000E+00
!NEAR   :  1.94342E+01
!FAR    :  3.88685E+01
CAME   2 EYE  -1.56209E+01  9.23770E+00  1.44814E+01 ! Tilted view
!        Q     9.00472E-01 -1.77386E-01 -3.93317E-01 -5.46453E-02
         VIEW  6.88956E-01 -3.62449E-01 -6.27671E-01
         RIGH  6.84631E-01  4.11253E-02  7.27729E-01
         UP    2.37951E-01  9.31096E-01 -2.76477E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  0.00000E+00  1.01980E+00  2.50000E-01
!RSPHERE:  6.47808E+00
!RADIUS :  2.26733E+01
!ASPECT :  1.00000E+00
!NEAR   :  1.61952E+01
!FAR    :  3.56294E+01
SCEN GEOM NAVI FREE
         !FACE HFRO
         !PINB PARE
    !VECT SCCO FIEL VITE SCAL USER PROG 10 PAS 10 140 TERM
     LIMA ON
SLER CAM1 2 NFRA 1
FREQ 1
!TRAC OFFS FICH AVI NOCL NFTO 41 FPS 15 KFRE 10 COMP -1 NFAI REND
TRAC OFFS FICH AVI NOCL NFTO 27 FPS 15 KFRE 10 COMP -1 NFAI REND
!GOTR LOOP 39 OFFS FICH AVI CONT NOCL NFAI REND
GOTR LOOP 25 OFFS FICH AVI CONT NOCL NFAI REND
GO
TRAC OFFS FICH AVI CONT NFAI REND
ENDPLAY
*=====================================================================
FIN
```

pith08b.epx

```
PITH08B
ECHO
 CONV WIN
OPTI PRIN
RESU ALIC 'pith08.ali' GARD PSCR
SORT VISU NSTO 1
*=====================================================================
PLAY
CAME   1 EYE   0.00000E+00  1.01980E+00  2.61623E+01 ! Front view
!        Q     1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
         VIEW  0.00000E+00  0.00000E+00 -1.00000E+00
         RIGH  1.00000E+00  0.00000E+00  0.00000E+00
         UP    0.00000E+00  1.00000E+00  0.00000E+00
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  0.00000E+00  1.01980E+00  2.50000E-01
!RSPHERE:  6.47808E+00
!RADIUS :  2.59123E+01
!ASPECT :  1.00000E+00
!NEAR   :  1.94342E+01
!FAR    :  3.88685E+01
CAME   2 EYE  -1.56209E+01  9.23770E+00  1.44814E+01 ! Tilted view
!        Q     9.00472E-01 -1.77386E-01 -3.93317E-01 -5.46453E-02
         VIEW  6.88956E-01 -3.62449E-01 -6.27671E-01
         RIGH  6.84631E-01  4.11253E-02  7.27729E-01
         UP    2.37951E-01  9.31096E-01 -2.76477E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  0.00000E+00  1.01980E+00  2.50000E-01
!RSPHERE:  6.47808E+00
!RADIUS :  2.26733E+01
```

```
!ASPECT :  1.00000E+00
!NEAR   :  1.61952E+01
!FAR    :  3.56294E+01
SCEN GEOM NAVI FREE
          FACE HFRO
         !LINE HEOU SSHA
          PINB PARE CDES
    !VECT SCCO FIEL VITE SCAL USER PROG 10 PAS 10 140 TERM
     LIMA ON
SLER CAM1 2 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 27 FPS 15 KFRE 10 COMP -1 NFAI REND
GOTR LOOP 25 OFFS FICH AVI CONT NOCL NFAI REND
GO
TRAC OFFS FICH AVI CONT NFAI REND
ENDPLAY
*=====================================================================
FIN
```

pith08c.epx

```
PITH08C
ECHO
 CONV WIN
OPTI PRIN
RESU ALIC 'pith08.ali' GARD PSCR
SORT VISU NSTO 1
*=====================================================================
PLAY
CAME   1 EYE   0.00000E+00  1.01980E+00  2.61623E+01 ! Front view
!        Q     1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
         VIEW  0.00000E+00  0.00000E+00 -1.00000E+00
         RIGH  1.00000E+00  0.00000E+00  0.00000E+00
         UP    0.00000E+00  1.00000E+00  0.00000E+00
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  0.00000E+00  1.01980E+00  2.50000E-01
!RSPHERE:  6.47808E+00
!RADIUS :  2.59123E+01
!ASPECT :  1.00000E+00
!NEAR   :  1.94342E+01
!FAR    :  3.88685E+01
CAME   2 EYE  -1.56209E+01  9.23770E+00  1.44814E+01 ! Tilted view
!        Q     9.00472E-01 -1.77386E-01 -3.93317E-01 -5.46453E-02
         VIEW  6.88956E-01 -3.62449E-01 -6.27671E-01
         RIGH  6.84631E-01  4.11253E-02  7.27729E-01
         UP    2.37951E-01  9.31096E-01 -2.76477E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  0.00000E+00  1.01980E+00  2.50000E-01
!RSPHERE:  6.47808E+00
!RADIUS :  2.26733E+01
!ASPECT :  1.00000E+00
!NEAR   :  1.61952E+01
!FAR    :  3.56294E+01
SCEN GEOM NAVI FREE
          FACE HFRO
         !LINE HEOU SSHA
          PINB CDES
    !VECT SCCO FIEL VITE SCAL USER PROG 10 PAS 10 140 TERM
     LIMA ON
SLER CAM1 2 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 27 FPS 15 KFRE 10 COMP -1 NFAI REND
GOTR LOOP 25 OFFS FICH AVI CONT NOCL NFAI REND
GO
TRAC OFFS FICH AVI CONT NFAI REND
ENDPLAY
*=====================================================================
FIN
```

pith08d.epx

```
PITH08D
ECHO
 CONV WIN
OPTI PRIN
RESU ALIC 'pith08.ali' GARD PSCR
SORT VISU NSTO 1
*=====================================================================
PLAY
CAME   1 EYE   0.00000E+00  1.01980E+00  2.61623E+01 ! Front view
!        Q     1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
         VIEW  0.00000E+00  0.00000E+00 -1.00000E+00
         RIGH  1.00000E+00  0.00000E+00  0.00000E+00
         UP    0.00000E+00  1.00000E+00  0.00000E+00
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  0.00000E+00  1.01980E+00  2.50000E-01
!RSPHERE:  6.47808E+00
!RADIUS :  2.59123E+01
!ASPECT :  1.00000E+00
!NEAR   :  1.94342E+01
!FAR    :  3.88685E+01
CAME   2 EYE  -1.56209E+01  9.23770E+00  1.44814E+01 ! Tilted view
!        Q     9.00472E-01 -1.77386E-01 -3.93317E-01 -5.46453E-02
         VIEW  6.88956E-01 -3.62449E-01 -6.27671E-01
         RIGH  6.84631E-01  4.11253E-02  7.27729E-01
         UP    2.37951E-01  9.31096E-01 -2.76477E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  0.00000E+00  1.01980E+00  2.50000E-01
!RSPHERE:  6.47808E+00
!RADIUS :  2.26733E+01
!ASPECT :  1.00000E+00
!NEAR   :  1.61952E+01
!FAR    :  3.56294E+01
SCEN GEOM NAVI FREE
          FACE HFRO
         !LINE HEOU SSHA
          PINB PARE
    !VECT SCCO FIEL VITE SCAL USER PROG 10 PAS 10 140 TERM
     LIMA ON
SLER CAM1 2 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 27 FPS 15 KFRE 10 COMP -1 NFAI REND
GOTR LOOP 25 OFFS FICH AVI CONT NOCL NFAI REND
GO
TRAC OFFS FICH AVI CONT NFAI REND
ENDPLAY
*=====================================================================
FIN
```

pith08z.epx

```
PITH08Z
ECHO
 CONV WIN
OPTI PRIN
RESU ALIC 'pith08.ali' GARD PSCR
SORT VISU NSTO 41
FIN
```

pith09.dgibi

```
opti echo 1;
opti titr 'pith09';
opti dime 3 elem cub8;
opti trac psc ftra 'pith09_mesh.ps';
opti sauv form 'pith09.msh';
*
p1 = -3 -3 0;
p2 =  3 -3 0;
p3 =  3  0 0;
p4 = -3  0 0;
c1 = p1 d 12 p2;
c2 = p2 d  6 p3;
c3 = p3 d 12 p4;
c4 = p4 d  6 p1;
bastar = dall c1 c2 c3 c4 plan;
target = bastar volu tran 1 (0 0 0.5);
*
gap = 0.1;
tol = 0.01;
*
p5 = -5  (5 + gap) 0;
p6 =  0  (0 + gap) 0;
p7 =  5  (5 + gap) 0;
p8 =  4  (5 + gap) 0;
p9 =  0  (1 + gap) 0;
p10= -4  (5 + gap) 0;
pc =  0  (5 + gap) 0;
c5 = p10 d 2 p5;
proj1 = c5 rota 16 90 pc (pc plus (0 0 1));
c6 = p9  d 2 p6;
proj2 = c6 rota 16 90 pc (pc plus (0 0 1));
baspro = proj1 et proj2;
elim tol (baspro et p7 et p8);
projec = baspro volu tran 1 (0 0 0.5);
*
mesh = target et projec;
*
tass mesh noop;
trac cach qual mesh;
*
sauv form mesh;
fin;
```

pith09.epx

```
PITH09
ECHO
!CONV WIN
EROS 1.0
CAST mesh
LAGR TRID
DIME ADAP NPOI 7000 CUBE 6000 NPIN 6000 ENDA TERM
GEOM CUBE target projec TERM
COMP COUL VERT LECT target TERM
          TURQ LECT projec TERM
     NGRO 1 'bloc' LECT target TERM COND Y LT -2.99
ADAP THRS ECRO 13 TMIN 0.1 TMAX 0.5 MAXL 3
          LECT projec TERM
MATE LINE RO 8000. YOUN 1.D11 NU 0.3
          LECT target TERM
DPDC RO 2360 YOUN 40.1E+9 NU 0.2
     FC 75.E+6 DAGG 16.0E-3  VERS 8 EFVI
     EROD ENDT 0.99 ENDC 0.99 DVOL 0.3
          LECT projec _cube TERM
```

```
OPTI PINS EQVL
LINK COUP SPLT NONE
     BLOQ 123 LECT bloc TERM
     PINB BODY MLEV 0 LECT target TERM
          SELF MLEV 0 LECT projec TERM
          ADAP LMAX 3 SCAL 1.0 SCAS 0.55
INIT VITE 2 -100 LECT projec TERM
ECRI DEPL VITE ACCE FINT FEXT FLIA FDEC CONT ECRO TFRE 0.005
     FICH ALIC TFRE 0.0005
OPTI PAS AUTO NOTE LOG 1
     CSTA 0.5
     ADAP RCON
     PINS GRID DPIN 1.01
CALC TINI 0. TEND 0.02D0 TFAI 3.E-6
FIN
```

pith09a.epx

```
PITH09A
ECHO
 CONV WIN
OPTI PRIN
RESU ALIC 'pith09.ali' GARD PSCR
SORT VISU NSTO 1
*=====================================================================
PLAY
CAME   1 EYE   0.00000E+00  1.01980E+00  2.61623E+01 ! Front view
!        Q     1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
         VIEW  0.00000E+00  0.00000E+00 -1.00000E+00
         RIGH  1.00000E+00  0.00000E+00  0.00000E+00
         UP    0.00000E+00  1.00000E+00  0.00000E+00
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  0.00000E+00  1.01980E+00  2.50000E-01
!RSPHERE:  6.47808E+00
!RADIUS :  2.59123E+01
!ASPECT :  1.00000E+00
!NEAR   :  1.94342E+01
!FAR    :  3.88685E+01
CAME   2 EYE  -1.56209E+01  9.23770E+00  1.44814E+01 ! Tilted view
!        Q     9.00472E-01 -1.77386E-01 -3.93317E-01 -5.46453E-02
         VIEW  6.88956E-01 -3.62449E-01 -6.27671E-01
         RIGH  6.84631E-01  4.11253E-02  7.27729E-01
         UP    2.37951E-01  9.31096E-01 -2.76477E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  0.00000E+00  1.01980E+00  2.50000E-01
!RSPHERE:  6.47808E+00
!RADIUS :  2.26733E+01
!ASPECT :  1.00000E+00
!NEAR   :  1.61952E+01
!FAR    :  3.56294E+01
SCEN GEOM NAVI FREE
         !FACE HFRO
         !PINB PARE
    !VECT SCCO FIEL VITE SCAL USER PROG 10 PAS 10 140 TERM
     LIMA ON
SLER CAM1 2 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 26 FPS 15 KFRE 10 COMP -1 NFAI REND
GOTR LOOP 24 OFFS FICH AVI CONT NOCL NFAI REND
GO
TRAC OFFS FICH AVI CONT NFAI REND
ENDPLAY
*=====================================================================
FIN
```

pith09b.epx

```
PITH09B
ECHO
 CONV WIN
OPTI PRIN
RESU ALIC 'pith09.ali' GARD PSCR
SORT VISU NSTO 1
*=====================================================================
PLAY
CAME   1 EYE   0.00000E+00  1.01980E+00  2.61623E+01 ! Front view
!        Q     1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
         VIEW  0.00000E+00  0.00000E+00 -1.00000E+00
         RIGH  1.00000E+00  0.00000E+00  0.00000E+00
         UP    0.00000E+00  1.00000E+00  0.00000E+00
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  0.00000E+00  1.01980E+00  2.50000E-01
!RSPHERE:  6.47808E+00
!RADIUS :  2.59123E+01
!ASPECT :  1.00000E+00
!NEAR   :  1.94342E+01
!FAR    :  3.88685E+01
CAME   2 EYE  -1.56209E+01  9.23770E+00  1.44814E+01 ! Tilted view
!        Q     9.00472E-01 -1.77386E-01 -3.93317E-01 -5.46453E-02
         VIEW  6.88956E-01 -3.62449E-01 -6.27671E-01
         RIGH  6.84631E-01  4.11253E-02  7.27729E-01
         UP    2.37951E-01  9.31096E-01 -2.76477E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  0.00000E+00  1.01980E+00  2.50000E-01
!RSPHERE:  6.47808E+00
!RADIUS :  2.26733E+01
!ASPECT :  1.00000E+00
!NEAR   :  1.61952E+01
!FAR    :  3.56294E+01
SCEN GEOM NAVI FREE
          FACE HFRO
         !LINE HEOU SSHA
          PINB PARE CDES
    !VECT SCCO FIEL VITE SCAL USER PROG 10 PAS 10 140 TERM
     LIMA ON
SLER CAM1 2 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 26 FPS 15 KFRE 10 COMP -1 NFAI REND
GOTR LOOP 24 OFFS FICH AVI CONT NOCL NFAI REND
GO
TRAC OFFS FICH AVI CONT NFAI REND
ENDPLAY
*=====================================================================
FIN
```

pith09c.epx

```
PITH09C
ECHO
 CONV WIN
OPTI PRIN
RESU ALIC 'pith09.ali' GARD PSCR
SORT VISU NSTO 1
*=====================================================================
PLAY
CAME   1 EYE   0.00000E+00  1.01980E+00  2.61623E+01 ! Front view
!        Q     1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
         VIEW  0.00000E+00  0.00000E+00 -1.00000E+00
         RIGH  1.00000E+00  0.00000E+00  0.00000E+00
         UP    0.00000E+00  1.00000E+00  0.00000E+00
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  0.00000E+00  1.01980E+00  2.50000E-01
!RSPHERE:  6.47808E+00
!RADIUS :  2.59123E+01
!ASPECT :  1.00000E+00
!NEAR   :  1.94342E+01
!FAR    :  3.88685E+01
CAME   2 EYE  -1.56209E+01  9.23770E+00  1.44814E+01 ! Tilted view
!        Q     9.00472E-01 -1.77386E-01 -3.93317E-01 -5.46453E-02
         VIEW  6.88956E-01 -3.62449E-01 -6.27671E-01
         RIGH  6.84631E-01  4.11253E-02  7.27729E-01
         UP    2.37951E-01  9.31096E-01 -2.76477E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  0.00000E+00  1.01980E+00  2.50000E-01
!RSPHERE:  6.47808E+00
!RADIUS :  2.26733E+01
!ASPECT :  1.00000E+00
!NEAR   :  1.61952E+01
!FAR    :  3.56294E+01
SCEN GEOM NAVI FREE
          FACE HFRO
         !LINE HEOU SSHA
          PINB CDES
    !VECT SCCO FIEL VITE SCAL USER PROG 10 PAS 10 140 TERM
     LIMA ON
SLER CAM1 2 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 26 FPS 15 KFRE 10 COMP -1 NFAI REND
GOTR LOOP 24 OFFS FICH AVI CONT NOCL NFAI REND
GO
TRAC OFFS FICH AVI CONT NFAI REND
ENDPLAY
*=====================================================================
FIN
```

pith09d.epx

```
PITH09D
ECHO
 CONV WIN
OPTI PRIN
RESU ALIC 'pith09.ali' GARD PSCR
SORT VISU NSTO 1
*=====================================================================
PLAY
CAME   1 EYE   0.00000E+00  1.01980E+00  2.61623E+01 ! Front view
!        Q     1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
         VIEW  0.00000E+00  0.00000E+00 -1.00000E+00
         RIGH  1.00000E+00  0.00000E+00  0.00000E+00
         UP    0.00000E+00  1.00000E+00  0.00000E+00
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  0.00000E+00  1.01980E+00  2.50000E-01
!RSPHERE:  6.47808E+00
!RADIUS :  2.59123E+01
!ASPECT :  1.00000E+00
!NEAR   :  1.94342E+01
!FAR    :  3.88685E+01
CAME   2 EYE  -1.56209E+01  9.23770E+00  1.44814E+01 ! Tilted view
!        Q     9.00472E-01 -1.77386E-01 -3.93317E-01 -5.46453E-02
         VIEW  6.88956E-01 -3.62449E-01 -6.27671E-01
         RIGH  6.84631E-01  4.11253E-02  7.27729E-01
         UP    2.37951E-01  9.31096E-01 -2.76477E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
```

```
!CENTER :  0.00000E+00  1.01980E+00  2.50000E-01
!RSPHERE:  6.47808E+00
!RADIUS :  2.26733E+01
!ASPECT :  1.00000E+00
!NEAR   :  1.61952E+01
!FAR    :  3.56294E+01
SCEN GEOM NAVI FREE
          FACE HFRO
         !LINE HEOU SSHA
          PINB PARE
    !VECT SCCO FIEL VITE SCAL USER PROG 10 PAS 10 140 TERM
     LIMA ON
SLER CAM1 2 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 26 FPS 15 KFRE 10 COMP -1 NFAI REND
GOTR LOOP 24 OFFS FICH AVI CONT NOCL NFAI REND
GO
TRAC OFFS FICH AVI CONT NFAI REND
ENDPLAY
*=================================================================
FIN
```

pith09z.epx

```
PITH09Z
ECHO
 CONV WIN
OPTI PRIN
RESU ALIC 'pith09.ali' GARD PSCR
SORT VISU NSTO 26
FIN
```

self01.epx

```
SELF01
ECHO
 CONV WIN
EROS 1.0
LAGR CPLA
DIME ADAP NPOI 10 CAR1 10 NPIN 10 ENDA TERM
GEOM LIBR POIN 9 CAR1 4 TERM
  0 0  1 0  2 0
  0 1  1 1  2 1
  0 2  1 2  2 2
  1 2 5 4
  2 3 6 5
  4 5 8 7
  5 6 9 8
COMP GROU 1 'square' LECT 1 PAS 1 4 TERM
     COUL ROSE LECT square TERM
MATE LINE RO 8000. YOUN 1.D11 NU 0.3
          LECT square _car1 TERM
OPTI PINS EQVL
LINK COUP SPLT NONE
     PINB SELF MLEV 0 LECT square TERM
          ADAP LMAX 3 SCAL 1.0 SCAS 1.0
ECRI DEPL VITE ACCE FINT FEXT FLIA FDEC CONT ECRO TFRE 0.005
     FICH ALIC TFRE 0.0005
OPTI PAS AUTO NOTE LOG 1
     CSTA 0.5
CALC TINI 0. TEND 0.D0 NMAX 0
PLAY
CAME   1 EYE   1.00000E+00  1.00000E+00  6.42907E+00
!        Q     1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
         VIEW  0.00000E+00  0.00000E+00 -1.00000E+00
         RIGH  1.00000E+00  0.00000E+00  0.00000E+00
         UP    0.00000E+00  1.00000E+00  0.00000E+00
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  1.00000E+00  1.00000E+00  0.00000E+00
!RSPHERE:  1.60727E+00
!RADIUS :  6.42907E+00
!ASPECT :  1.00000E+00
!NEAR   :  4.82180E+00
!FAR    :  9.64361E+00
SCEN GEOM NAVI FREE
          FACE HFRO
          PINB PARE
SLER CAM1 1 NFRA 1
TRAC OFFS FICH BMP REND
ENDPLAY
FIN
```

self02.epx

```
SELF02
ECHO
 CONV WIN
EROS 1.0
LAGR CPLA
GEOM LIBR POIN 16 CAR1 9 TERM
  0 0  1 0  2 0  3 0
  0 1  1 1  2 1  3 1
  0 2  1 2  2 2  3 2
  0 3  1 3  2 3  3 3
  1 2 6 5
  2 3 7 6
  3 4 8 7
  5 6 10 9
  6 7 11 10
  7 8 12 11
  9 10 14 13
  10 11 15 14
  11 12 16 15
COMP GROU 1 'square' LECT 1 PAS 1 9 TERM
     COUL ROSE LECT square TERM
MATE LINE RO 8000. YOUN 1.D11 NU 0.3
          LECT square _car1 TERM
OPTI PINS EQVL
LINK COUP SPLT NONE
     PINB SELF MLEV 0 LECT square TERM
ECRI DEPL VITE ACCE FINT FEXT FLIA FDEC CONT ECRO TFRE 0.005
     FICH ALIC TFRE 0.0005
OPTI PAS AUTO NOTE LOG 1
     CSTA 0.5
CALC TINI 0. TEND 0.D0 NMAX 0
PLAY
CAME   1 EYE   1.50000E+00  1.50000E+00  9.13165E+00
!        Q     1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
         VIEW  0.00000E+00  0.00000E+00 -1.00000E+00
         RIGH  1.00000E+00  0.00000E+00  0.00000E+00
         UP    0.00000E+00  1.00000E+00  0.00000E+00
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  1.50000E+00  1.50000E+00  0.00000E+00
!RSPHERE:  2.28291E+00
!RADIUS :  9.13165E+00
!ASPECT :  1.00000E+00
!NEAR   :  6.84874E+00
!FAR    :  1.36975E+01
SCEN GEOM NAVI FREE
          FACE HFRO
          PINB PARE
SLER CAM1 1 NFRA 1
TRAC OFFS FICH BMP REND
ENDPLAY
FIN
```

self03.epx

```
SELF03
ECHO
 CONV WIN
EROS 1.0
LAGR CPLA
DIME ADAP NPOI 144 CAR1 160 NPIN 160 ENDA TERM
GEOM LIBR POIN 16 CAR1 9 TERM
  0 0  1 0  2 0  3 0
  0 1  1 1  2 1  3 1
  0 2  1 2  2 2  3 2
  0 3  1 3  2 3  3 3
  1 2 6 5
  2 3 7 6
  3 4 8 7
  5 6 10 9
  6 7 11 10
  7 8 12 11
  9 10 14 13
  10 11 15 14
  11 12 16 15
COMP GROU 1 'square' LECT 1 PAS 1 9 TERM
     COUL ROSE LECT square TERM
MATE LINE RO 8000. YOUN 1.D11 NU 0.3
          LECT square _car1 TERM
OPTI PINS EQVL
LINK COUP SPLT NONE
     PINB SELF MLEV 0 LECT square TERM
          ADAP LMAX 3 SCAL 1.0 SCAS 1.0
ECRI DEPL VITE ACCE FINT FEXT FLIA FDEC CONT ECRO TFRE 0.005
     FICH ALIC TFRE 0.0005
OPTI PAS AUTO NOTE LOG 1
     CSTA 0.5
CALC TINI 0. TEND 0.D0 NMAX 0
PLAY
CAME   1 EYE   1.50000E+00  1.50000E+00  9.13165E+00
!        Q     1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
         VIEW  0.00000E+00  0.00000E+00 -1.00000E+00
         RIGH  1.00000E+00  0.00000E+00  0.00000E+00
         UP    0.00000E+00  1.00000E+00  0.00000E+00
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  1.50000E+00  1.50000E+00  0.00000E+00
!RSPHERE:  2.28291E+00
!RADIUS :  9.13165E+00
!ASPECT :  1.00000E+00
!NEAR   :  6.84874E+00
!FAR    :  1.36975E+01
SCEN GEOM NAVI FREE
          FACE HFRO
          PINB PARE
SLER CAM1 1 NFRA 1
TRAC OFFS FICH BMP REND
ENDPLAY
FIN
```

self04.epx

```
SELF04
ECHO
 CONV WIN
EROS 1.0
LAGR CPLA
DIME ADAP NPOI 144 CAR1 160 NPIN 160 ENDA TERM
GEOM LIBR POIN 16 CAR1 9 TERM
  0 0  1 0  2 0  3 0
  0 1  1 1  2 1  3 1
  0 2  1 2  2 2  3 2
  0 3  1 3  2 3  3 3
  1 2 6 5
  2 3 7 6
  3 4 8 7
  5 6 10 9
  6 7 11 10
  7 8 12 11
  9 10 14 13
  10 11 15 14
  11 12 16 15
COMP GROU 1 'square' LECT 1 PAS 1 9 TERM
     COUL ROSE LECT square TERM
MATE LINE RO 8000. YOUN 1.D11 NU 0.3
          LECT square _car1 TERM
OPTI PINS EQVL
LINK COUP SPLT NONE
     PINB SELF MLEV 0 LECT square TERM
          ADAP LMAX 3 SCAL 0.880 SCAS 0.880
ECRI DEPL VITE ACCE FINT FEXT FLIA FDEC CONT ECRO TFRE 0.005
     FICH ALIC TFRE 0.0005
OPTI PAS AUTO NOTE LOG 1
     CSTA 0.5
CALC TINI 0. TEND 0.DO NMAX 0
PLAY
CAME   1 EYE   1.50000E+00  1.50000E+00  9.13165E+00
!        Q     1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
         VIEW  0.00000E+00  0.00000E+00 -1.00000E+00
         RIGH  1.00000E+00  0.00000E+00  0.00000E+00
         UP    0.00000E+00  1.00000E+00  0.00000E+00
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  1.50000E+00  1.50000E+00  0.00000E+00
!RSPHERE:  2.28291E+00
!RADIUS :  9.13165E+00
!ASPECT :  1.00000E+00
!NEAR   :  6.84874E+00
!FAR    :  1.36975E+01
SCEN GEOM NAVI FREE
          FACE HFRO
          PINB PARE
SLER CAM1 1 NFRA 1
TRAC OFFS FICH BMP REND
ENDPLAY
FIN
```

self05.epx

```
SELF05
ECHO
 CONV WIN
EROS 1.0
LAGR CPLA
DIME ADAP NPOI 144 CAR1 160 NPIN 160 ENDA TERM
GEOM LIBR POIN 16 CAR1 9 TERM
  0 0  1 0  2 0  3 0
  0 1  1 1  2 1  3 1
  0 2  1 2  2 2  3 2
  0 3  1 3  2 3  3 3
  1 2 6 5
  2 3 7 6
  3 4 8 7
  5 6 10 9
  6 7 11 10
  7 8 12 11
  9 10 14 13
  10 11 15 14
  11 12 16 15
COMP GROU 1 'square' LECT 1 PAS 1 9 TERM
     COUL ROSE LECT square TERM
MATE LINE RO 8000. YOUN 1.D11 NU 0.3
          LECT square _car1 TERM
OPTI PINS EQVL
LINK COUP SPLT NONE
     PINB SELF MLEV 0 LECT square TERM
          ADAP LMAX 3 SCAL 0.890 SCAS 0.890
ECRI DEPL VITE ACCE FINT FEXT FLIA FDEC CONT ECRO TFRE 0.005
     FICH ALIC TFRE 0.0005
OPTI PAS AUTO NOTE LOG 1
     CSTA 0.5
CALC TINI 0. TEND 0.DO NMAX 0
PLAY
CAME   1 EYE   1.50000E+00  1.50000E+00  9.13165E+00
!        Q     1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
         VIEW  0.00000E+00  0.00000E+00 -1.00000E+00
         RIGH  1.00000E+00  0.00000E+00  0.00000E+00
         UP    0.00000E+00  1.00000E+00  0.00000E+00
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  1.50000E+00  1.50000E+00  0.00000E+00
!RSPHERE:  2.28291E+00
!RADIUS :  9.13165E+00
!ASPECT :  1.00000E+00
!NEAR   :  6.84874E+00
!FAR    :  1.36975E+01
SCEN GEOM NAVI FREE
          FACE HFRO
          PINB PARE
SLER CAM1 1 NFRA 1
TRAC OFFS FICH BMP REND
ENDPLAY
FIN
```

List of input files

LB-NA-27888-EN-N

JRC Mission

As the Commission's in-house science service, the Joint Research Centre's mission is to provide EU policies with independent, evidence-based scientific and technical support throughout the whole policy cycle.

Working in close cooperation with policy Directorates-General, the JRC addresses key societal challenges while stimulating innovation through developing new methods, tools and standards, and sharing its know-how with the Member States, the scientific community and international partners.

Serving society
Stimulating innovation
Supporting legislation

doi:10.27888/333017

ISBN 978-92-79-57972-1